

# NAVAL POSTGRADUATE SCHOOL

Monterey, California

AD-A247 521



## THESIS

AN ANALYSIS OF LIGHT INFANTRY  
EFFECTIVENESS IN MID-TO-HIGH INTENSITY CONFLICT  
DELIBERATE ATTACK MISSIONS

by

Steven J. Hutchison

June 1991

Thesis Advisor:

Michael P. Bailey

Approved for public release; distribution is unlimited

92 8 10 100

92-06839



Unclassified

security classification of this page

## REPORT DOCUMENTATION PAGE

1a Report Security Classification <b>Unclassified</b>			1b Restrictive Markings		
2a Security Classification Authority			3 Distribution Availability of Report		
2b Declassification Downgrading Schedule			Approved for public release; distribution is unlimited.		
4 Performing Organization Report Number(s)			5 Monitoring Organization Report Number(s)		
6a Name of Performing Organization <b>Naval Postgraduate School</b>		6b Office Symbol (if applicable) <b>OR</b>	7a Name of Monitoring Organization <b>Naval Postgraduate School</b>		
6c Address (city, state, and ZIP code) <b>Monterey, CA 93943-5000</b>			7b Address (city, state, and ZIP code) <b>Monterey, CA 93943-5000</b>		
8a Name of Funding Sponsoring Organization		8b Office Symbol (if applicable)	9 Procurement Instrument Identification Number		
8c Address (city, state, and ZIP code)			10 Source of Funding Numbers		
			Program Element No	Project No	Task No
			Work Unit Accession No		
11 Title (include security classification) <b>AN ANALYSIS OF LIGHT INFANTRY EFFECTIVENESS IN MID-TO-HIGH INTENSITY CONFLICT DELIBERATE ATTACK MISSIONS</b>					
12 Personal Author(s) <b>Steven J. Hutchison</b>					
13a Type of Report <b>Master's Thesis</b>		13b Time Covered From To		14 Date of Report (year, month, day) <b>1991, June</b>	15 Page Count <b>129</b>
16 Supplementary Notation The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.					
17 Cosati Codes			18 Subject Terms (continue on reverse if necessary and identify by block number)		
Field	Group	Subgroup	National Training Center Heavy/Light Rotation, Simulation		
19 Abstract (continue on reverse if necessary and identify by block number)					
<p>This thesis documents a simulation study of light infantry operations in mid-to-high intensity conflict. An initial data analysis is performed using deliberate attack missions conducted at the U.S. Army National Training Center (NTC) and compares the measures of effectiveness (MOE) of fully modernized heavy forces to the effectiveness of heavy forces operating with an attached light infantry battalion. This analysis includes development of a light infantry attack simulation which employs object oriented programming in MODSIM II. The simulation models light infantry operations in the NTC environment and is used to explore alternative tactical employment techniques designed to enhance unit performance on the AirLand Battlefield. This thesis also describes the tank and mechanized infantry task force, the light infantry task force, the heavy/light rotation concept, the deliberate attack mission, and the NTC environment and data collection capabilities.</p> <p>The simulation models an infantry attack against opposing forces in fixed, fortified positions. The model is a high resolution simulation which builds object code from infantry platoon level through battalion. The simulation depicts unit movements, attrition to indirect fires, and target engagements. The positioning of enemy forces is extracted from actual battlefield positions during an NTC deliberate attack mission. The simulation replicates close operations in which the light force mission is to gain an initial penetration of enemy barriers and pass the heavy force forward to continue the attack. The simulation study explores the use of light forces in alternative tactical scenarios.</p>					
20 Distribution Availability of Abstract			21 Abstract Security Classification		
<input checked="" type="checkbox"/> unclassified unlimited <input type="checkbox"/> same as report <input type="checkbox"/> DTIC users			<b>Unclassified</b>		
22a Name of Responsible Individual <b>Michael P. Bailey</b>			22b Telephone (include Area code) <b>(408) 646-2085</b>		22c Office Symbol <b>OR, Ba</b>

Approved for public release; distribution is unlimited.

An Analysis of Light Infantry Effectiveness in  
Mid-to-High Intensity Conflict Deliberate Attack Missions

by

Steven J. Hutchison  
Captain, United States Army  
B.S., United States Military Academy, 1982

Submitted in partial fulfillment of the  
requirements for the degree of

MASTER OF SCIENCE IN OPERATIONS RESEARCH

from the

NAVAL POSTGRADUATE SCHOOL  
June 1991

Author:

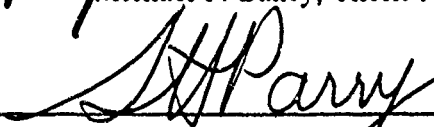


Steven J. Hutchison

Approved by:



Michael P. Bailey, Thesis Advisor



Samuel H. Parry, Second Reader



Peter Purdue, Chairman,  
Department of Operations Research

## ABSTRACT

This thesis documents a simulation study of light infantry operations in mid-to-high intensity conflict. An initial data analysis is performed using deliberate attack missions conducted at the U.S. Army National Training Center (NTC) and compares the measures of effectiveness (MOE) of fully modernized heavy forces to the effectiveness of heavy forces operating with an attached light infantry battalion. This analysis includes development of a light infantry attack simulation which employs object oriented programming in MODSIM II. The simulation models light infantry operations in the NTC environment and is used to explore alternative tactical employment techniques designed to enhance unit performance on the AirLand Battlefield. This thesis also describes the tank and mechanized infantry task force, the light infantry task force, the heavy/light rotation concept, the deliberate attack mission, and the NTC environment and data collection capabilities.

The simulation models an infantry attack against opposing forces in fixed, fortified positions. The model is a high resolution simulation which builds object code from infantry platoon level through battalion. The simulation depicts unit movements, attrition to indirect fires, and target engagements. The positioning of enemy forces is extracted from actual battlefield positions during an NTC deliberate attack mission. The simulation replicates close operations in which the light force mission is to gain an initial penetration of enemy barriers and pass the heavy force forward to continue the attack. The simulation study explores the use of light forces in alternative tactical scenarios.



Accession For	
DTIC GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By	
Distribution/	
Availability Codes	
Dist	Avail and/or Special
A-1	

## TABLE OF CONTENTS

I. INTRODUCTION .....	1
A. BACKGROUND .....	1
B. PURPOSE AND SCOPE .....	2
C. SCENARIO .....	3
1. General .....	3
a. Phase I - Deployment .....	3
b. Phase II - Light Infantry Operations .....	3
2. The Light Infantry Attack Simulation .....	3
3. Future Developments .....	3
D. PROBLEM DESCRIPTION .....	4
II. HEAVY LIGHT PERFORMANCE ASSESSMENT .....	5
A. GENERAL .....	5
B. TASK ORGANIZATION .....	5
1. The Light Infantry Battalion .....	5
a. Employment of Light Infantry .....	5
b. Organization of the Light Infantry Battalion .....	5
2. The Tank and Mechanized Infantry Task Force .....	6
C. THE DELIBERATE ATTACK MISSION .....	6
D. NTC DATA COLLECTION .....	6
1. The NTC Environment .....	6
2. Mission of the NTC .....	6
3. Heavy/Light Rotation Description .....	9
4. Data Collection .....	9
E. DATA SOURCES .....	9
F. DELIBERATE ATTACK DATA .....	10
G. MEASURES OF EFFECTIVENESS .....	11
1. General .....	11
2. Destroy MOE .....	11
3. Survival MOE .....	12
H. ANALYSIS .....	14

III. LIGHT INFANTRY ATTACK SIMULATION .....	18
A. PURPOSE .....	18
1. Mission Planning .....	18
2. Battle Analysis .....	18
B. MODEL PROGRAMMING .....	18
C. MODEL EXECUTION .....	19
1. Force Representation .....	19
2. Execution .....	19
D. MODEL DESIGN .....	21
1. Model Components .....	21
a. Attack .....	21
b. Globals .....	21
c. Unit .....	21
d. Firepower Capability (FPC) .....	22
e. ATGM .....	22
f. Map Reconnaissance .....	22
g. OPFOR .....	22
h. Impact .....	23
i. Weapons .....	23
j. Artillery .....	23
k. MOE .....	23
l. Menu .....	23
2. Use of Random Number Generators .....	24
E. MODEL CAPABILITIES .....	24
1. Movement .....	24
a. Position Identification .....	24
b. Distance .....	25
c. Movement Time .....	25
2. Direct Fires .....	25
a. Engagement Aspect Angle .....	26
b. Damage Assessment .....	28
3. Indirect Fires .....	28
4. Attrition .....	29
5. Target Assignments, Reassignments, and Target Handover. ....	30
a. Target Assignments .....	30

b. Reassignment .....	30
c. Target Handover .....	31
F. MODEL INPUT .....	31
1. Scenario Input .....	31
a. Forces .....	31
b. Concept of the Operation .....	32
2. Model Parameters .....	32
G. MODEL OUTPUT .....	32
IV. SIMULATION ANALYSIS .....	33
A. GENERAL .....	33
B. OUTPUT ANALYSIS .....	33
C. THE BASELINE MODEL .....	33
1. NTC Heavy/Light Mission AA89xxxx .....	34
a. Battle Replay with GNATT II .....	34
2. Scenario Input .....	34
a. Scheduling of Indirect Fires .....	34
b. Forces .....	36
3. Baseline Model Results .....	36
D. A REAR ATTACK PLAN .....	38
1. Concept of the Operation .....	38
2. Model Results .....	38
E. A FLANK ATTACK PLAN .....	38
1. Concept of the Operation .....	38
2. Model Results .....	38
F. COMPARISON OF RESULTS .....	39
V. CONCLUSIONS AND RECOMMENDATIONS .....	44
A. CONCLUSIONS .....	44
B. RECOMMENDATIONS .....	44
APPENDIX A. MODSIM CODE .....	46
A. ATTACK .....	46
B. GLOBALS .....	48
C. UNIT .....	51

D. FPC .....	65
E. ATGM .....	73
F. MAPRECON .....	79
G. OPFOR .....	83
H. IMPACT .....	86
I. WEAPONS .....	90
J. ARTY .....	93
K. MOE .....	95
L. MENU .....	98
 APPENDIX B. INPUT DATA FILES .....	 101
A. OPFOR DATA .....	101
B. MISSILE DATA .....	101
C. P-KILL DATA .....	101
D. TRANSPORTATION DATA .....	102
 APPENDIX C. SCENARIO INPUT .....	 103
A. BASELINE MODEL .....	103
B. REAR ATTACK MODEL .....	104
C. FLANK ATTACK MODEL .....	105
 APPENDIX D. SAMPLE ATTRITION OUTPUT .....	 106
 APPENDIX E. SAMPLE ENGAGEMENT HISTORY .....	 108
 APPENDIX F. BASELINE MODEL OUTPUT .....	 109
A. RESULTS OF 500 REPLICATIONS .....	109
B. RESULTS OF A TRIAL WITHOUT ARTILLERY .....	109
C. RESULTS OF A TRIAL WITH ARTILLERY .....	110
 APPENDIX G. REAR ATTACK MODEL OUTPUT .....	 111
A. RESULTS OF 500 REPLICATIONS .....	111
B. RESULTS OF A TRIAL WITHOUT ARTILLERY .....	111
C. RESULTS OF A TRIAL WITH ARTILLERY .....	112



APPENDIX.H. FLANK ATTACK MODEL OUTPUT .....	113
A. RESULTS OF 500 REPLICATIONS .....	113
B. RESULTS OF A TRIAL RUN WITHOUT ARTILLERY .....	113
C. RESULTS OF A TRIAL RUN WITH ARTILLERY .....	114
LIST OF REFERENCES .....	115
INITIAL DISTRIBUTION LIST .....	117

## LIST OF TABLES

Table 1.	HEAVY LIGHT OPFOR DESTRUCTION DATA .....	12
Table 2.	HEAVY FORCE OPFOR DESTRUCTION DATA .....	13
Table 3.	HEAVY LIGHT FRIENDLY SURVIVAL DATA .....	14
Table 4.	HEAVY FORCE FRIENDLY SURVIVAL DATA .....	15
Table 5.	RESULTS OF CRN TEST OF DIFFERENCE .....	43

## LIST OF FIGURES

Figure 1. The Mechanized Infantry Task Force .....	7
Figure 2. The National Training Center .....	8
Figure 3. Friendly Force Organization in the Model .....	20
Figure 4. Gun-Target relationship .....	27
Figure 5. Impact Area Designation .....	29
Figure 6. GNATT II Replay of AA89xxxx .....	35
Figure 7. Baseline Model Operations Overlay .....	37
Figure 8. Rear Attack Operations Overlay .....	39
Figure 9. Flank Attack Operations Overlay .....	40
Figure 10. Confidence Intervals for Each Scenario .....	41
Figure 11. Scenario Distributions .....	42

## I. INTRODUCTION

### A. BACKGROUND

The Army of the 21st Century must meet the fundamental requirements of versatility, deployability, and lethality. AirLand Battle, the Army's current doctrine, provides the framework for organizing, training, and equipping forces to maximize combat power and effectiveness across the spectrum of conflict, from low to high intensity. The requirement to maintain an appropriate mix of heavy, light, and special operations forces is one of six Army fundamental imperatives [Ref. 1]. At the operational and tactical levels, heavy and light forces must be prepared to fight on an integrated battlefield to exploit and optimize the capabilities of each force. The Army's Combat Training Centers (CTCs) provide a tough, realistic environment in which to train forces to fight on the combined arms battlefield.

The National Training Center (NTC) is the testbed of the AirLand Battle doctrine. One of the recurrent themes of training at the NTC is the integration of light infantry forces on the battlefield with heavy force operations. The first of the "heavy, light" rotations was conducted in the spring of 1987. There have been 12 heavy, light rotations out of some 66 rotations at the time of this writing. The application of combined heavy and light force operations stems from the AirLand Battle imperatives, which are fundamental for success on the modern battlefield. Specifically, the one imperative which describes the purpose of integrating forces is entitled "Combine Arms and Sister Services to Complement and Reinforce." Complementary combined arms expose the enemy to the effects of one arm while he attempts to evade the effects of another. Arms and services reinforce each other when one serves to increase the effectiveness of the other or combine to produce mass. [Ref. 2: p. 25]

Successful integration of light and heavy forces is a combat multiplier on the battlefield. Intuitively, an analysis of units fighting as part of a combined arms force should suggest a measurable increase in the effectiveness of engaged forces. The NTC provides the data collection environment to test such an hypothesis. However, after reviewing observer comments over numerous heavy, light rotations, an apparent trend seems evident: our heavy and light forces are not synchronized in their efforts on the battlefield. Additionally, a cursory analysis of battlefield damage and casualty rates indicates that the light forces typically contribute little to the overall battle while suffering

overwhelming casualties. Major General Peter J. Boylan, in a recent article addressing the employment of heavy and light forces in mid-to-high intensity conflict, states

It has been demonstrated time and time again that, other conditions being equal, light forces pitted against heavy combat forces will suffer unacceptably high losses or be defeated almost 100 percent of the time. The defeat of enemy heavy maneuver forces will almost certainly require the employment of similar type heavy forces, even with enhanced light force technology. [Ref. 3: p. 28]

## **B. PURPOSE AND SCOPE**

The purpose of this thesis is to analyze light infantry effectiveness through development and experimentation with a simulation model. The scope of this thesis is limited to modeling light force operations in a heavy/light scenario consistent with the capabilities of the NTC. This thesis emphasizes the development and use of a light force simulation model and employs the object oriented programming language MODSIM II.

Current Army capabilities to test light infantry operations in a mid-to-high intensity environment are limited. Analyzing light infantry effectiveness may be accomplished through several approaches, two of which are presented here: an analysis of measures of effectiveness in training at the CTCs, and simulating light infantry operations in a combat model. The Army's CTCs provide a training environment in which light forces are routinely employed; however, light forces are seldom used in a role which maximizes their utility in this type of environment. Furthermore, US Army combat models are also limited in their ability to model light infantry operations.

Light infantry performance at the NTC is difficult to measure quantitatively. NTC battles typically focus on the destruction of enemy maneuver forces as opposed to other elements of enemy combat power. Heavy/light battles are a graphic manifestation of this shortcoming. Commanders rarely have the opportunity to employ light forces against enemy battlefield operating systems other than their heavy maneuver forces, resulting in unacceptably high losses. There are numerous factors which influence the ability of the light force to accomplish its mission. Some of the factors are readily obtained from the comments of the observers, while others are so intermixed with the performance of the entire heavy/light force as to render them intangible. Those quantifiable factors will be used to perform the initial data analysis and determine measures of effectiveness. Chapter II discusses light infantry performance at the NTC. However, due to limited ability of the CTCs to provide scenarios in which the light force's combat power may be maximized, and limited data availability, further analysis via simulation methodology may provide insight into improving light infantry effectiveness.

## C. SCENARIO

### 1. General

The general scenario portrays a requirement to commit friendly forces in rugged, open terrain, against a modern armored and mechanized opposing force. The friendly force mission is to conduct a deliberate attack to seize objectives and destroy enemy forces. The friendly force is organized around a fully modernized tank and mechanized infantry task force and light infantry forces. The commander's intent is to insert the light force early to penetrate barriers and fix the enemy front line. The heavy forces will exploit the penetration and attack deep into enemy territory as the main effort.

#### *a. Phase I - Deployment*

The first phase of the model, deployment of forces, assumes successful insertion of the light forces, either to forward positions in front of the enemy, or to position to the enemy's flank or rear. No actual modeling is performed; this simply provides a starting position from which the light forces begin ground operations.

#### *b. Phase II - Light Infantry Operations*

The second phase is modeled by the light infantry attack simulation. During this phase, light infantry elements are operating against enemy fixed, fortified positions. Enemy positions and weapons systems are extracted from actual battlefield positions during an NTC deliberate attack mission, and several friendly courses of action form the basis of the experiment.

### 2. The Light Infantry Attack Simulation

The model is a high resolution combat simulation which discretely represents the infantry battalion, rifle companies, rifle platoons, and each Anti-Tank Guided Missile (ATGM) gunner in the platoon. The simulation depicts unit movements, attrition to indirect fires, and target engagements. The simulation permits employment of light forces in alternative tactical situations. Further discussion of the model is contained in Chapter III.

### 3. Future Developments

The natural extension of this effort is the development of a complementary heavy force model or analytic surrogate. Continued development of scenarios to allow simultaneous employment of light and heavy forces in complementary force operations would enable a more complete analysis of the total force effectiveness.

#### **D. PROBLEM DESCRIPTION**

The approach of this thesis is to construct and analyze, by simulation methodology, a model to explore alternative light infantry tactics in a mid-to-high intensity deliberate attack scenario. The NTC heavy/light rotation deliberate attack missions provide a data source for determining measures of effectiveness and employment characteristics.

This thesis is an initial effort to simulate light force operations. It is both timely and relevant; planning successful complementary operations pose a significant problem to tactical units preparing to fight on an integrated battlefield. This research and analysis may be used to enhance the battle staff planning process and tactical execution, and to provide doctrinal insight into methods to achieve results that neither force could achieve operating on its own.

## II. HEAVY/LIGHT PERFORMANCE ASSESSMENT

### A. GENERAL

This Chapter presents an assessment of heavy/light training and performance at the NTC. Data collected from numerous heavy and heavy/light rotations are presented to further define the problem. The motivation for this presentation is to:

- Provide data input to the light infantry attack simulation (discussed further in Chapter IV).
- Determine the measures of effectiveness to analyze unit performance.
- Highlight shortcomings in the ability of the C I Cs to support complementary force operations, instrumentation, and performance evaluations.
- Describe shortcomings in heavy/light tactics.

### B. TASK ORGANIZATION

#### 1. The Light Infantry Battalion

##### *a. Employment of Light Infantry*

Infantry units have the unique quality of being an all-weather force capable of defeating any enemy on any terrain. Infantry is ideally suited for close-in operations against an enemy of equal mobility, or in terrain which degrades the mobility of mechanized forces [Ref. 4]. In operations where armored forces predominate, infantry can:

- Make initial penetrations in difficult terrain for exploitation by armor and mechanized infantry.
- Attack over approaches that are not feasible for heavy forces.
- Conduct rear area operations, capitalizing on air mobility. [Ref. 2: p. 41]

##### *b. Organization of the Light Infantry Battalion*

An infantry battalion consists of a headquarters, maneuver units, combat support (CS), and combat service support (CSS) elements. The battalion is typically augmented with additional CS and CSS assets based on the mission, enemy, terrain, troops and time available (METT-T). The maneuver forces organic to the infantry battalion include three rifle companies and one anti-armor company. Normal augmentation to the battalion includes a fire support battery, engineers, air defense, and other elements.



## **2. The Tank and Mechanized Infantry Task Force**

The heavy battalion task force is organized by cross-attaching tank and mechanized infantry companies within the brigade. A battalion task force usually consists of four to five maneuver companies, an anti-armor company, a headquarters element, and various slices of CS and CSS assets. An example of a mechanized infantry task force is shown in Figure 1.

## **C. THE DELIBERATE ATTACK MISSION**

The deliberate attack mission is the most detailed and thoroughly coordinated offensive mission for the battle staff planner. For the tactical unit, the deliberate attack is the most difficult and challenging to execute. All elements of combat power are brought to bear on the enemy. The deliberate attack is defined as:

An attack planned and carefully coordinated with all concerned elements based on thorough reconnaissance, evaluation of all available intelligence and relative combat strength, analysis of various courses of action and other factors affecting the situation. It generally is conducted against a well organized defense when a hasty attack cannot be conducted or has been conducted and failed. [Ref. 5 : p. 1-8]

Deliberate attacks are planned in detail, and are characterized by timely intelligence, extensive preparations, deception, electronic warfare, unconventional warfare, and psychological operations. Deep operations play a significant role in the deliberate attack. Deep operations are conducted to "block movement of [enemy] reserves, destroy his command posts, neutralize his artillery, and prevent the escape of targeted elements." [Ref. 2: p. 116] The deliberate attack is therefore selected as the focus for the study of heavy/light effectiveness and data collection.

## **D. NTC DATA COLLECTION**

### **1. The NTC Environment**

The NTC is located in the Mojave Desert at Fort Irwin, California. The NTC is a vast expanse of widely varying desert terrain covering some 640,000 acres. The mountainous terrain divides the maneuver area into three corridors; the northern corridor is used principally for live fire training while the central and southern corridors are used for force on force maneuver exercises. The training center is depicted in Figure 2.

### **2. Mission of the NTC**

The NTC has two primary missions. The first mission is to provide tough, realistic combined arms and joint services training in accordance with AirLand Battle doctrine, for brigades and regiments in a mid-to-high intensity environment, while

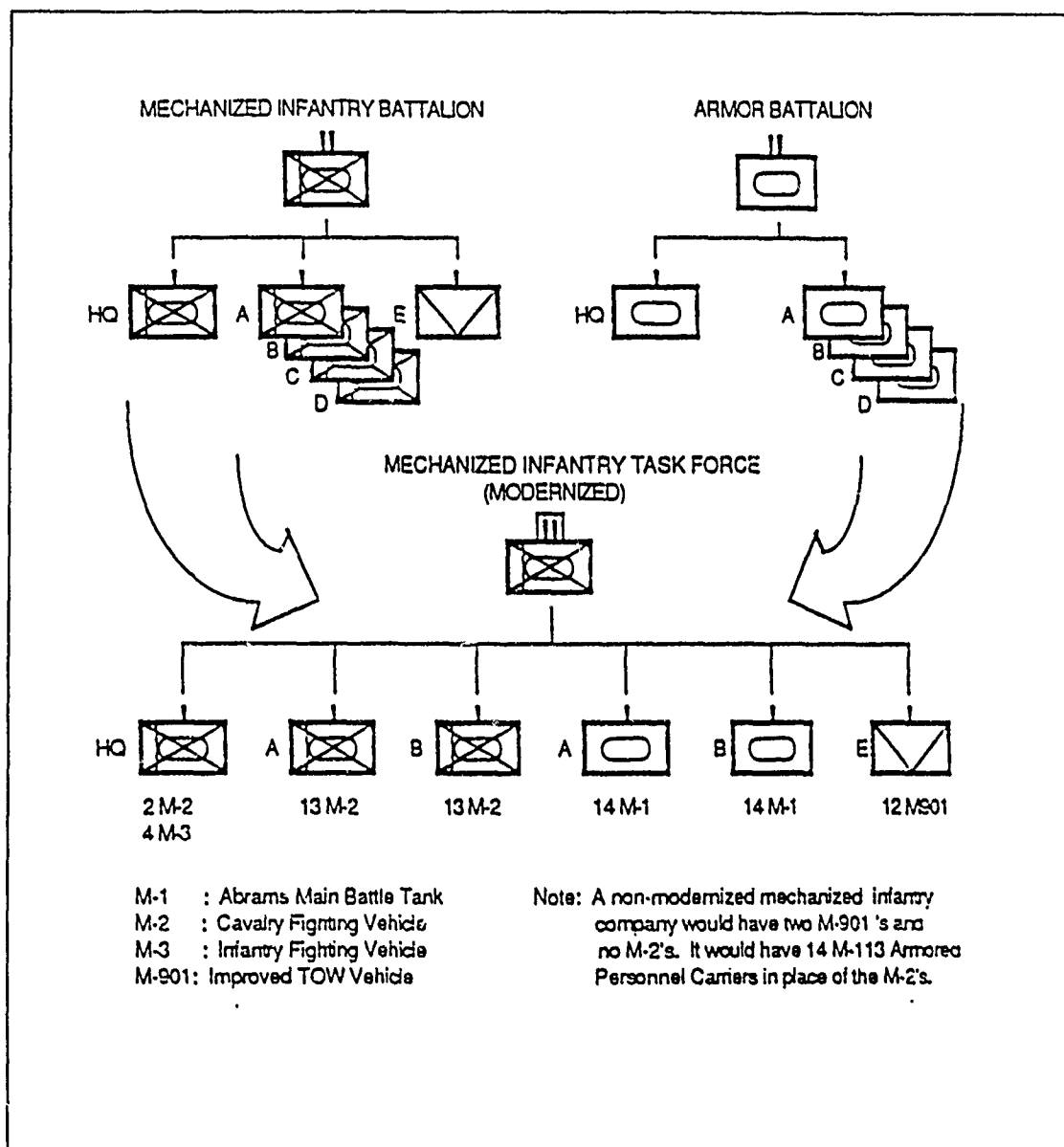


Figure 1. The Mechanized Infantry Task Force

retaining feedback and analysis at the battalion/task force level. The second mission is to provide a data source for training, doctrine, and equipment improvements. Training exercises are "free-play", allowing units to plan and fight as they would in combat, subject to specific safety guidelines and rules of engagement. Following each mission, units receive immediate performance feedback in the form of after action reviews (AARs). The AAR is a forum for commanders and staffs to evaluate their own

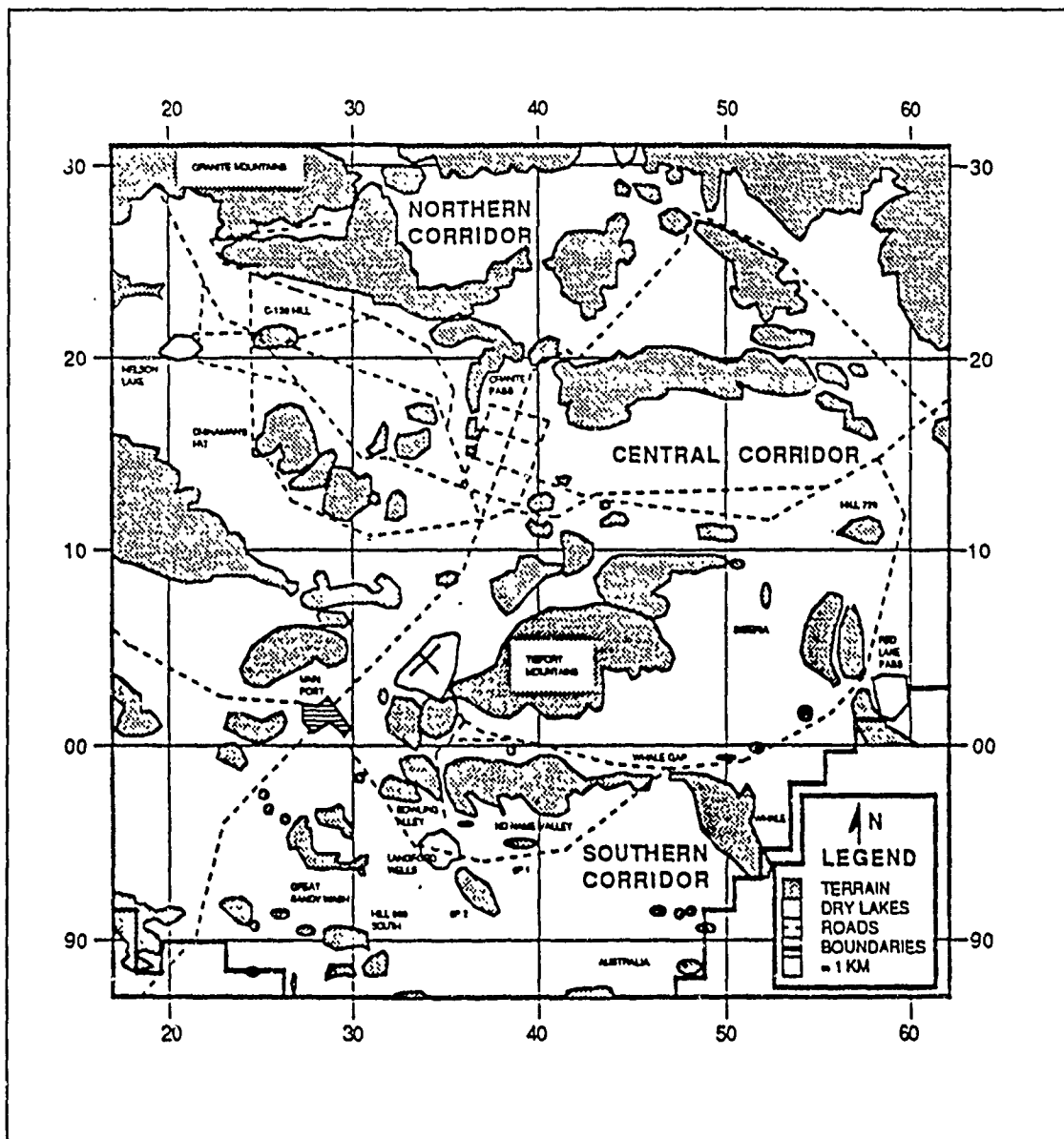


Figure 2. The National Training Center

performance and learn from comments of outside observers. The AAR focuses the analysis on the seven battlefield operating systems (BOS):

- Maneuver
- Command and Control
- Fire Support
- Intelligence

- Air Defense
- Mobility, Countermobility, Survivability
- Combat Service Support.

### **3. Heavy/Light Rotation Description**

Training at the NTC is conducted on a rotational basis. There are 14 rotations scheduled during each fiscal year. Forces deploying to the NTC typically consist of a brigade headquarters, two battalions of armor and/or mechanized infantry, an artillery battalion, and a support battalion. During a heavy/light rotation, a light infantry battalion from another division is attached to the heavy force brigade commander for the entire rotation.

The NTC has a permanently assigned opposing force (OPFOR) which is organized to replicate a Soviet style motorized rifle regiment, consisting of three motorized rifle battalions. OPFOR equipment consists of U.S. Army tracked and wheeled vehicles visually modified to more closely resemble threat equipment. The OPFOR is proficient in Soviet tactics, knows the terrain, and is highly motivated. There is no change in threat tactics when a heavy/light rotation is scheduled as compared to normal heavy rotations.

A typical rotation is divided into three phases: battalion force-on-force training (FFT), battalion live fire training (LFT), and brigade FFT. During a heavy/light rotation, the FFT and LFT usually consist of both heavy and light task forces operating under brigade control.

### **4. Data Collection**

The NTC's instrumentation system is the principal asset in collecting kill data and determining the source of the engagement. Player units are instrumented with the Multiple Integrated Laser Engagement System (MILES) down to vehicle level, enabling the mainframe subsystems to determine vehicle locations and status, resolve direct fire engagements, and to store the data for future analysis. Instrumentation of the light infantry battalion is less accurate. Although each individual soldier wears the MILES harness, and all light infantry weapons systems have a MILES firing device, position locating devices track only to platoon level, and casualty data are collected by observer/controllers moving with the units.

## **E. DATA SOURCES**

The Army Research Institute--Presidio of Monterey (ARI--POM), houses the CTC archive. The facility consists of the digital data archive, a non-digital data archive, and the Combat Operations Research Facility. The archives store all the data collected

during a rotation at the NTC. The data take many forms, including digital, audio-visual, written, and operations graphics.

The digital archive database provides the user rapid access to unit organizations, equipment composition, battle damage statistics, and battle replay. Digital data may be accessed through the VAX computer network or personal computer. The non-digital data archive stores written copies of the unit take-home packages, operations orders and overlays, video tape copies of the AARs, and audio recordings of radio transmissions. The data presented in this thesis represent a collection effort using all of these media, with emphasis on the unit take home packages.

The unit take home packages contain a written summary of the mission and include detailed comments from the observers relating unit performance in each of the seven BOS. Additionally, the take home package contains a statistical summary describing the casualty assessment of both friendly and OPFOR units, and identifies weapon systems that caused the casualties for each mission during the rotation. Kills of OPFOR systems attributable to light infantry actions are sometimes difficult to isolate; the statistical summary identifies systems that caused OPFOR casualties, but does not necessarily identify the unit that caused the casualties. An example of this data isolation problem is the TOW anti-tank guided missile: both light infantry forces and mechanized infantry forces engage tanks with TOWs. Determining which unit scored the hit under these conditions involves double-checking times and locations of unit engagements, and verifying weapons assigned to the unit.

#### **F. DELIBERATE ATTACK DATA**

The initial data analysis effort involved the collection of data to compare the operational effectiveness of heavy forces versus the effectiveness of heavy forces operating with an attached light infantry battalion. Data were collected for all deliberate attack missions conducted by fully modernized heavy forces operating without light infantry; i.e., units equipped with the M1 Abrams Main Battle Tank and the M2 Bradley Infantry Fighting Vehicle, and all deliberate attack missions in which light infantry operated in conjunction with heavy forces. A total of 26 heavy modernized deliberate attack missions were selected, with 14 heavy/light deliberate attack missions available. However, it must be noted that due to the small sample size of fully modernized heavy/light rotations (only six of the 14 available), all heavy/light deliberate attack missions were considered. Of the 26 heavy deliberate attack missions, seven were conducted by mechanized task forces, 13 by armor task forces, and six by brigade level units. The

heavy, light deliberate attacks included seven by armor heavy, five by mechanized infantry heavy, and two by brigade level forces.

## G. MEASURES OF EFFECTIVENESS

### 1. General

There are numerous factors to consider as measures of a unit's effectiveness. Some factors such as force ratios and number of systems destroyed are easily quantified, while other factors such as technology or leadership are not. The MOEs established here reflect, to some extent, the limitations of data availability and quantifiability. The statistical summary in the unit take home packages is the most reliable data source for collection. There are two aspects of operational planning which lead to quantifiable MOEs: destruction of the enemy force and protection of the friendly force. However, inconsistencies in the data prevent accurate analysis of all systems contributing to the effectiveness of the unit. In particular, the data tend to focus on the major tank killing systems and lack specificity and sufficient detail to accurately depict infantry losses. Therefore, the systems selected for study include the systems for which the data is most consistent: tanks, infantry fighting vehicles, and TOWs for the friendly force; tanks, BMPs, the Soviet equivalent of the M2, and BRDMs, the Soviet armored reconnaissance vehicle.

### 2. Destroy MOE

The first MOE, termed the Destroy MOE, is calculated as the total number of enemy systems killed during the attack divided by the total number of enemy systems at the start of the attack, for each observation  $i$ :

$$\text{Destroy MOE}_i = \frac{\text{OPFOR}(\text{Tanks}_i + \text{BMPs}_i + \text{BRDMs}_i)_{\text{destroyed}}}{\text{OPFOR}(\text{Tanks}_i + \text{BMPs}_i + \text{BRDMs}_i)_{\text{starting}}}$$

The data collected for heavy, light deliberate attacks and calculation of the Destroy MOE are shown in Table 1. Modernized heavy, light rotations are indicated by an asterisk in the rotation column. The combined efforts of the heavy and light forces achieved a mean destruction of 48.5% of the opposing forces, with a standard deviation of 21.6%. The range of destruction values is from 16% to approximately 84%. Table 2 contains the data for the heavy force deliberate attack missions. For the 26 heavy force observations, the mean destruction of opposing forces is 49.5%, with a standard deviation of 23.1%. Destruction of opposing forces ranged from 15.8% to approximately 96%. In terms of enemy destruction, heavy forces achieved a slightly higher level

of destruction than did the heavy/light forces operating in concert, contrary to the expected result. Further discussion of this result is presented in the analysis at the conclusion of this chapter.

Table 1. HEAVY/LIGHT OPFOR DESTRUCTION DATA

NO.	TANK		BMP		BRDM		Destroy MOE
	Start	Lost	Start	Lost	Start	Lost	
1	10	4	21	17	2	1	0.6667
2	4	4	14	8	2	1	0.6500
3	6	1	16	3	2	2	0.2500
4	14	6	23	4	4	1	0.2683
5	12	8	10	3	4	2	0.5000
6	13	4	23	9	4	2	0.3750
7	19	4	32	7	4	2	0.2364
8	13	10	29	18	5	2	0.6383
9	28	22	63	39	4	2	0.6632
10	19	1	20	5	4	7	0.3023
11	13	9	29	15	5	2	0.5532
12	16	13	33	28	7	6	0.8393
13	21	2	24	4	5	2	0.1600
14	16	10	18	15	5	2	0.6923

### 3. Survival MOE

The second MOE, termed the Survival MOE, is calculated as the total number of friendly systems surviving the attack divided by the total number of friendly systems at the start of the attack, for each observation  $i$ :

$$\text{Survival MOE}_i = \frac{\text{Friendly}(Tanks_i + BFV's_i + TOW's_i)\text{surviving}}{\text{Friendly}(Tanks_i + BFV's_i + TOW's_i)\text{starting}}$$

The data collected for the heavy/light deliberate attacks and calculation of the Survival MOE is shown in Table 3. The combined heavy and light forces achieved a mean survival rate of 32.3% of starting forces, with a standard deviation of only 9.4%. In this case, the range of friendly force survival is from 21% to approximately 51%. Table 4 contains the data for the heavy force deliberate attack missions. The heavy forces

Table 2. HEAVY FORCE OPFOR DESTRUCTION DATA

NO.	TANK		BMP		BRDM		Destroy MOE
	Start	Lost	Start	Lost	Start	Lost	
1	8	2	10	6	0	0	0.4444
2	8	1	18	7	0	0	0.3077
3	35	30	15	11	3	2	0.8113
4	8	7	15	8	0	0	0.6522
5	8	8	16	16	1	0	0.9600
6	38	17	81	30	6	5	0.4160
7	8	2	16	6	2	2	0.3846
8	14	11	44	31	2	2	0.7333
9	4	3	14	10	3	2	0.7143
10	12	2	30	9	3	1	0.2667
11	18	9	32	21	2	2	0.6154
12	22	5	38	9	4	1	0.2344
13	22	11	49	14	2	0	0.3425
14	13	10	35	25	3	3	0.7451
15	39	5	26	19	5	3	0.3857
16	21	7	38	9	2	0	0.2623
17	6	6	18	16	3	2	0.8889
18	6	4	18	13	3	2	0.7037
19	13	7	21	16	3	2	0.6757
20	19	3	34	5	4	1	0.1579
21	39	8	53	29	4	1	0.3958
22	19	8	49	15	6	2	0.3378
23	15	3	16	4	10	2	0.2195
24	18	4	35	10	10	0	0.2222
25	13	6	25	17	10	6	0.6042
26	17	5	24	10	6	4	0.4043

achieved a mean survival of 23.3% of friendly forces, with a standard deviation of 13.6%, and ranged from total force losses to 57% survival. In terms of friendly survival, the combined heavy/light forces obtained slightly higher protection than did heavy forces



operating alone. Note, however, that these data do not reflect the losses to light infantry; only armor systems are considered.

Table 3. HEAVY/LIGHT FRIENDLY SURVIVAL DATA

NO.	TANK		BFV/APC		ITV/TOW		Survival MOE
	Start	Lost	Start	Lost	Start	Lost	
1	27	9	23	12	24	13	0.5135
2	74	49	18	5	28	21	0.3750
3	31	20	44	26	13	6	0.4091
4	27	26	59	39	10	4	0.2813
5	26	24	57	28	10	8	0.2473
6	24	17	58	32	13	7	0.4105
7	27	22	26	18	17	13	0.2429
8	25	21	31	26	31	13	0.3103
9	52	32	64	58	29	21	0.2345
10	24	24	56	24	12	8	0.3913
11	26	23	56	43	28	21	0.2091
12	43	37	70	31	39	22	0.4079
13	28	27	20	12	4	2	0.2115
14	21	17	29	18	25	19	0.2800

## II. ANALYSIS

As stated, only those quantifiable measures of effectiveness were considered. Of significant importance in the heavy/light analysis is the absence of loss figures for dismounted infantry. This is an unfortunate consequence of the limitations of the NTC to collect data which adequately reflect the quantity and cause of infantry losses, and inconsistencies in the data that do exist.

There are several factors that are not considered when analyzing the data from a purely start/loss perspective. Comments from the observer/controllers (OCs), which observe and evaluate each mission, provide valuable insight into the apparent inability of the heavy and light forces to achieve a measurable increase in effectiveness on the integrated battlefield.

An equipment shortcoming directly affects the ability of the light force to achieve kills against OPI/OR armored equipment in the NTC environment. The primary light

Table 4. HEAVY FORCE FRIENDLY SURVIVAL DATA

NO.	TANK		BFV		ITV		Survival MOE
	Start	Lost	Start	Lost	Start	Lost	
1	15	14	38	22	8	3	0.3607
2	38	21	23	18	4	4	0.3385
3	39	26	19	14	3	0	0.3443
4	26	18	21	20	0	0	0.1915
5	23	16	28	22	10	7	0.2623
6	47	27	62	34	13	6	0.4508
7	25	14	31	27	10	4	0.3182
8	45	44	55	47	7	4	0.1121
9	26	22	21	19	5	4	0.1346
10	36	29	16	7	6	3	0.3276
11	75	60	37	24	10	3	0.2869
12	38	38	16	16	0	0	0.000
13	22	16	32	23	11	4	0.3385
14	61	26	49	24	11	2	0.5702
15	60	57	51	36	10	7	0.1736
16	32	28	20	18	0	0	0.1154
17	22	16	25	15	11	8	0.3276
18	21	17	25	22	10	10	0.1250
19	24	20	28	26	0	0	0.1154
20	11	9	41	34	8	4	0.2167
21	33	31	71	54	9	5	0.2035
22	38	26	26	18	0	0	0.3125
23	20	20	22	13	3	1	0.2444
24	21	21	24	21	3	3	0.0625
25	20	19	23	23	3	3	0.0217
26	29	28	26	21	0	0	0.1091

infantry anti-tank system is the Dragon missile, a man portable, optically tracked, wire guided missile, designed to defeat most enemy armor. The Dragon has a MILES counterpart; the standard day tracker has an integrated MILES firing device. Infantry

Dragon gunners also train to engage targets at night using the AN TAS-5 Thermal Night Sight. However, there is currently no night vision device with integrated MILES; hence, infantry units operating during limited visibility conditions are unable to kill armored targets, which significantly reduces their ability to contribute to the battle in a measureable sense.

Another frequent OC comment is the susceptibility of light infantry to the effects of indirect fires. Once detected, artillery frequently renders the infantry ineffective before the dismounted force reaches the objective. The terrain of the NTC offers little cover to the effects of indirect fires.

The typical *modus operandi* of an NTC heavy/light deliberate attack is a product of the NTC training environment. Without targetable OPFOR battlefield operating systems other than the maneuver forces, the friendly commander frequently resorts to tasking the light infantry battalion to attack, under cover of darkness, to seize an initial foothold in the enemy defenses, breach obstacles, and establish lanes through which the heavy force will pass to maintain the momentum of the attack. Such attacks are typically frontal, the least desirable form of maneuver in the deliberate attack. Not only does this method employ the light force against an enemy it is not designed to defeat, given the terrain, it is further complicated when combining forces not accustomed to each other's capabilities, limitations, and standard operating procedures. Frequent OC comments indicate that the heavy/light deliberate attack increases the overall complexity of the operation, as suggested by operational problems ranging from land navigation, failure of the light force to gain the initial foothold, unrehearsed recognition signals, friendly fire casualties resulting from the light force presence in the objective area, and loss of momentum at the passage point. Clausewitz, the oft cited military theoretician, might have described this as the 'fog of war' [Ref. 6].

Unquestionably, the major objective of the friendly force is the destruction of the enemy's maneuver elements. However, with the introduction of light forces into the organization,

...the legitimacy of such an approach comes into question. We have proved over and over that in a confrontation between light and heavy combat forces, in other than close terrain, light forces incur a significant disadvantage. Nonetheless, because of the inability of our training centers to provide a scenario that incorporates the cumulative impact of indirect attacks on combat support, CSS and command and control throughout the depth of the battlefield, light forces are generally required to be employed in a manner which ill suits their utility in such an environment. [Ref. 3: pp. 31-32]

The opportunity for the light force to attack enemy BOS and conduct deep operations does not exist. The introduction of light forces provides the means to attack the enemy in depth while concentrating their efforts against enemy elements they are capable of defeating. Employed in this context, the simultaneity of attack by heavy and light forces poses a dilemma for the enemy, which is a fundamental element of successful complementary force operations.

### **III. LIGHT INFANTRY ATTACK SIMULATION**

#### **A. PURPOSE**

##### **1. Mission Planning**

The light infantry attack simulation provides a useful planning tool to prepare units for operations in mid-to-high intensity conflict. As a planning device, the simulation model allows the battle staff planner to simulate various courses of action developed in the planning process, and to predict outcomes. The light infantry attack model emphasizes intelligence and operations estimates. The intent of the simulation is to enable exploration of various courses of action based on the current estimate of the enemy situation, assist in the decision making process, and examine light infantry doctrine in a mid-to-high intensity environment.

##### **2. Battle Analysis**

The simulation model can also be employed as a training analysis tool. The CTC data archives provide the input information so that results of actual CTC battles can be compared to simulated outcomes. The simulation can be designed to replicate CTC battles to assist in evaluating unit performance. Additionally, as a training device, the user can compare results of alternative courses of action with those of the actual battle plan.

#### **B. MODEL PROGRAMMING**

MODSIM II is a general purpose, modular programming language which provides highly portable, object-oriented programming and discrete event simulation [Ref. 7]. The modular concept adds flexibility in programming and encapsulates objects which can then be imported for use in other programs. Modules consist of three types: definition, implementation, and a main module. Definition modules contain a set of definitions for export to other modules; implementation modules contain the actual code for executing the defined methods. A main module is the only required module, and contains the routine of the program.

MODSIM II provides dynamic allocation of objects, records, and arrays. Objects contain fields and methods; methods contain a sequence of instructions which manipulate the object's variable fields. ASK METHODS are synchronous methods, and do not elapse simulation time when executed. TELL METHODS are asynchronous, time elapsing sets of instructions, which when implemented, are placed on the simulation

calendar and executed in time sequence. PROCEDURES are another construct which perform computations and other instructions in similar fashion as subroutines in other programming languages.

### C. MODEL EXECUTION

#### 1. Force Representation

The light infantry attack simulation represents both friendly and enemy forces in object code. The friendly forces are hierarchically organized from battalion down to ATGM level, while the opposing forces are represented as a series of distinct objects arrayed on the battlefield. Figure 3 depicts the friendly force organization in the attack simulation.

Friendly forces consist of a battalion headquarters and three rifle companies, each consisting of three rifle platoons. The rifle platoon is uniquely defined in two components: the platoon headquarters and the elements of the platoon's firepower capability. The headquarters executes unit activities, such as movement or message passing. The firepower capability (FPC), also defined in object code, executes individual soldier activities, such as firing. The FPC discretely represents the major anti-armor systems in an infantry platoon, while maintaining a numerical accounting for the sum of all remaining elements, including leaders, riflemen, automatic riflemen, grenadiers, and machine gunners.

#### 2. Execution

Once compiled, MODSIM II creates an executable file with the same name as the main module. The model is executed simply by invoking the name of the model. This is another feature which contributes to the exportability of MODSIM programs. The light infantry attack simulation is executed with the command *Attack*. A brief description of the flow of the model follows.

The model begins and queries the user to select the tactical experiment, which is coordinated to a particular input file. The choices include execution of the baseline model, a flank attack model, or a rear attack model. The scenarios are discussed in more detail in Chapter IV. A second menu provides the user the opportunity to conduct a "walk-through" of the model or to replicate an input number of iterations. The walk-through writes output comments to the screen for the user to observe as the model progresses. A selection to replicate will prompt the user to input the number of iterations, run without output to the screen, collect the critical data, and write this

calendar and executed in time sequence. PROCEDURES are another construct which perform computations and other instructions in similar fashion as subroutines in other programming languages.

## C. MODEL EXECUTION

### 1. Force Representation

The light infantry attack simulation represents both friendly and enemy forces in object code. The friendly forces are hierarchically organized from battalion down to ATGM level, while the opposing forces are represented as a series of distinct objects arrayed on the battlefield. Figure 3 depicts the friendly force organization in the attack simulation.

Friendly forces consist of a battalion headquarters and three rifle companies, each consisting of three rifle platoons. The rifle platoon is uniquely defined in two components: the platoon headquarters and the elements of the platoon's firepower capability. The headquarters executes unit activities, such as movement or message passing. The firepower capability (FPC), also defined in object code, executes individual soldier activities, such as firing. The FPC discretely represents the major anti-armor systems in an infantry platoon, while maintaining a numerical accounting for the sum of all remaining elements, including leaders, riflemen, automatic riflemen, grenadiers, and machine gunners.

### 2. Execution

Once compiled, MODSIM II creates an executable file with the same name as the main module. The model is executed simply by invoking the name of the model. This is another feature which contributes to the exportability of MODSIM programs. The light infantry attack simulation is executed with the command *Attack*. A brief description of the flow of the model follows.

The model begins and queries the user to select the tactical experiment, which is coordinated to a particular input file. The choices include execution of the baseline model, a flank attack model, or a rear attack model. The scenarios are discussed in more detail in Chapter IV. A second menu provides the user the opportunity to conduct a "walk-through" of the model or to replicate an input number of iterations. The walk-through writes output comments to the screen for the user to observe as the model progresses. A selection to replicate will prompt the user to input the number of iterations, run without output to the screen, collect the critical data, and write this

company completes the initial attack, the company moves to an intermediate objective from which to reengage targets which were not destroyed. The model terminates when all companies have completed the final assault and either all targets are destroyed or the unit is out of ammunition.

#### **D. MODEL DESIGN**

##### **1. Model Components**

The light infantry attack simulation consists of 23 modules: 11 definition, 11 implementation, and one main module. The simulation code is contained in Appendix A. A brief description of the principal components of the model follows:

##### **a. Attack**

Main module *Attack* sets the routine of the program. The module imports several procedures to setup the background data for the program. These procedures include setting the seeds for the random number generators, reading the transportation and missile system parameters, modeling the enemy defense, and reading the data to model the unit operations plan. Additionally, the main module creates the battalion and all subordinate units, implements the battalion object's TELL METHOD *ExecuteMission*, and starts the simulation.

##### **b. Globals**

The *Globals* definition and implementation modules include selected variables which may be seen throughout the program. These variables include the random number generators, and variables defining characteristics of the battlefield, such as visibility condition, weapons status, and transportation data. The implementation module sets the values of these variables at run-time, allocates the random number generators, and opens the output files.

##### **c. Unit**

The *Unit* modules provide the structure for each unit object in the model. Units consist of four levels of unit objects: *UnitObj*, *RiflePlatoonObj*, *RifleCompanyObj*, and *BattalionObj*. The generic *UnitObj* defines fields and methods common to all units, and are inherited by each of the other specific units. The methods of each of the units define events which normally occur at unit level, such as movement along a specified route, occupation of firing positions, and target assignments. The battalion object has, in addition to its other fields, a trigger object. The trigger object provides a means of synchronizing events in the simulation. An identification field is attached to each unit; companies are named A, B, and C. Additionally, within companies, for example,



platoons are further identified as A1, A2, or A3. The use of identifiers is a critical asset when viewing the model in progress and reviewing output files.

*d. Firepower Capability (FPC)*

The *FPC* modules are an element of the model architecture whose purpose is to define methods focused at the soldier level. The separation between platoon events and events within the *FPCObj*, is a means of encapsulating events occurring at squad, fireteam, and individual soldier level. The *FPCObj* performs the numerical accounting of each subordinate element within a rifle platoon's firepower capability. Additionally, the *FPC* discretely allocates the platoon's ATGM gunners as object variables. The methods perform both accountability and message passing. The *FPCObj* also has an identification, which corresponds exactly with the parent platoon.

*e. ATGM*

The *ATGM* modules detail the direct fire capability of the unit. The *ATGMObj* contains in its fields the missile data read at the start of the program. The methods detail the engagement sequence of an ATGM gunner and include preparation of the round, target acquisition, tracking, and assessment of target damage. Additionally, the ATGM objects contain a trigger mechanism, which grants permission to the gun to fire based on current weapons status or once the synchronized attack commences. Each ATGM is also given an identity. The ATGM identity consists of the platoon (*FPC*) to which it belongs, appended with the number assigned to each system, either one or two. For example, an ATGM identity of A12 signifies A company, first platoon, second ATGM gunner.

*f. Map Reconnaissance*

The *MapRecon* modules contain a procedure to read a user constructed data file which is built during the planning process or to reconstruct a battle. Additionally, *MapRecon* allocates records to store positional information, and connects them in a linked list to form the unit movement routes. The *MapRecon* module also contains the *Distance* procedure. *Distance* takes as input arguments, two locations in UTM grid coordinates (six-digit, 100 meter coordinates with two letter grid zone identifier), and determines the straight line distance between the two points.

*g. OPFOR*

The *OPFOR* modules explicitly define each *OPFOR* vehicle on the battlefield as an *EnemyVehicleObj*. The *ModelEnemyDefense* procedure reads data from an input file, creates each *OPFOR* system, and assigns each system the input attributes.

As a planning tool, the user inputs data based on available intelligence; as an analytic device, the user inputs actual data gathered from the various sources.

#### *h. Impact*

The *Impact* module contains two procedures: a procedure to determine the engagement aspect angle, and a procedure to assess damage to a target. The procedure *AspectAngle* employs a vector mathematics formulation to determine the angle between the gun-target vector and the target orientation vector. The result of the procedure call is a determination of where on the target the round impacted as either front, flank, or rear. This information is passed to the *AssessDamage* procedure which performs a Monte Carlo draw on a random number generator, compares the sample to the missile system's probability of kill for that target and impact point combination, and returns the assessment of whether the target is killed or damaged.

#### *i. Weapons*

The *Weapons* modules contain two procedures to read the specific weapon system characteristics and the probability of kill data. The user supplies the data for the program to read from a data file. The kill probability data used in this model are merely approximations of actual data under similar conditions. The data include an estimate of the probability of kill for a Dragon missile versus four different OPFOR vehicles in frontal, flank, and rear engagements.

#### *j. Artillery*

The *Arty* modules define the procedure *ScheduleOPFORArtillery* which computes the probability of kill of the OPFOR artillery against the light forces. The model employs a Confetti approximation, assuming the light forces are uniformly distributed throughout a given target area. The artillery play is scheduled at run-time, based on the user's estimate of when movement will be compromised, and upon execution of the attack. The artillery model is currently the only means of causing attrition of the friendly forces.

#### *k. MOE*

The *MOE* modules provide continuous running means and variances on the measure of effectiveness for destruction of enemy forces. Upon termination of the run, the critical statistical data is written to an output file. In addition to maintaining the Destroy MOE, the model computes the mean mission time and mean level of attrition.

#### *l. Menu*

The *Menu* modules increase the utility of the model by prompting the user to select the particular scenario to be run, and then querying the user to select an option

to replicate, walk-through the model with artillery play, or walk-through the model without artillery play. Note that the selection to replicate always runs the model with scheduled artillery. A selection to walk-through the model either with or without artillery allows the user to observe unit movements, occurrence of the artillery strikes, and results of each engagement. A selection to replicate further prompts the user to input the number of replications, and the model runs without providing comments to the screen. position such that the platoon can engage, move, and engage again.

## **2. Use of Random Number Generators**

The light infantry attack simulation uses four distinct random number generators. The use of separate random number generators ensures comparability between multiple runs of the simulation, and is one of many techniques of variance reduction [Ref. 8: p. 47]. Random number generators are provided for sampling missile hit probabilities, probabilities of kill against vehicular targets, indirect fire losses, and selection of the number of rounds per gun fired in an artillery barrage.

## **E. MODEL CAPABILITIES**

The light infantry attack model simulates unit movements, direct and indirect fire engagements, force attrition, and target assignments. A general description of the algorithms used to implement these capabilities follows.

### **1. Movement**

The movement algorithm is a time-elapsing method common to all *Unit* objects. There are two key elements of the *MoveTo* method: identification of the destination, and determination of the movement time, which requires a measurement of the distance involved.

#### **a. Position Identification**

Positional information in the light infantry attack simulation is stored in a *RECORD* data structure. A record is dynamically allocated, contains variable fields, and differs from an object in that it has no methods which operate on its fields. A record can contain a reference variable of another record, thus facilitating construction of linked lists. Position records store the doctrinal name of the position, such as ATK PSN, a six digit center of mass grid coordinate (with two letter identifier) for the location, the locations of firing positions, if any, and a reference variable which points to the next position record along the unit's route. In this way, units may be "told" to move to the next position, with all the required information attached.

*b. Distance*

Computing the distance between two points given in grid coordinates is subject to the constraint that the two points will lie in two regions covered by adjacent UTM grid zone identifiers (this includes diagonally adjacent regions). The *Distance* procedure, defined in the *MapRecon* module, first compares the grid zone identifiers and then "normalizes" the relationship between the two points. The computation is then an application of the Pythagorean Theorem, and the resulting distance is returned in meters. For example, to find the distance between NK900150 and NL030200, the algorithm first compares NK to NL and identifies the points as lying in horizontally adjacent grid zones. The algorithm then normalizes the easterly coordinate 030 (interpreted as 3.0 kilometers) to 1030 so that the subtraction  $1030 - 900$  yields a horizontal change in distance of 13.0 kilometers. The vertical change is 5.0 kilometers, yielding a distance of 13,928.4 meters.

*c. Movement Time*

On implementation of *MoveTo*, the algorithm sets the field value for the movement start time as the current simulation time. The movement rate,  $R$ , is computed as

$$R_{ij} = CF_i MR_{ij} ,$$

where  $i$  = transportation type,  $j$  = visibility condition.  $MR$  is a matrix of movement rates, and  $CF$  is an array of conversion factors to convert movement rates given in knots or kilometers hour to meters sec. Movement rates are input based on data obtained from appropriate FMs. For example, Table 16-14, FM 5-34, gives the rate of march of infantry troops, cross-country, at night, as 1.6 km/hr [Ref. 9]. Movement time,  $T$ , is then computed using the standard formula  $T = D/R$ , where distance,  $D$ , in meters, is obtained from a call to the distance procedure. The method then waits the indicated time to move, and then updates ' ' 's position to the new position. In a situation where movement is interrupted, as during an artillery strike, the algorithm computes the amount of time remaining to complete the move, adds an arbitrary constant regroup time, and waits the remaining time before updating the unit's location.

**2. Direct Fires**

Direct fires in the light infantry attack simulation model only the major anti-armor systems organic to light infantry units. Only the Dragon anti-tank guided missile system is modeled; however, the *ATGMObj* is intended to be generic to both Dragon and

TOW systems. Direct fire engagements model preparation of the missile, acquisition of the target (range checking), firing, tracking and damage assessment, and taking the launcher out of action. Each stage in the engagement sequence is a method of the *ATGMObj*. The methods model the engagement sequence; determination of the result of the engagement is a two step process which involves computation of the engagement aspect angle and damage assessment.

*a. Engagement Aspect Angle*

The engagement aspect angle is determined by a call to the procedure *Aspect.Angle* contained in the *Impact* module. The aspect angle formula employed in the procedure is a result of the following derivation. The engagement aspect angle,  $\alpha$ , defined as the angle between the gun-target vector  $\vec{G}$  and the target orientation vector  $\vec{T}$ , is obtained from the formula

$$\cos \alpha = \frac{\vec{G} \cdot \vec{T}}{\|\vec{G}\| \|\vec{T}\|} . \quad (1)$$

To determine the engagement aspect angle, an arbitrary coordinate system is established such that the target location identifies the origin, and grid north (GN) defines  $0^\circ$ . Define

$$\gamma \equiv \text{gun-target angle}$$

and

$$\theta \equiv \text{target orientation angle,}$$

and let

$$g_1 \equiv \text{the horizontal component of } \vec{G} ,$$

$$g_2 \equiv \text{the vertical component of } \vec{G} ,$$

$$t_1 \equiv \text{the horizontal component of } \vec{T} ,$$

and

$$t_2 \equiv \text{the vertical component of } \vec{T} .$$

Figure 4a depicts the angular relationship of each system. Figure 4b depicts the components of the gun-target vector.

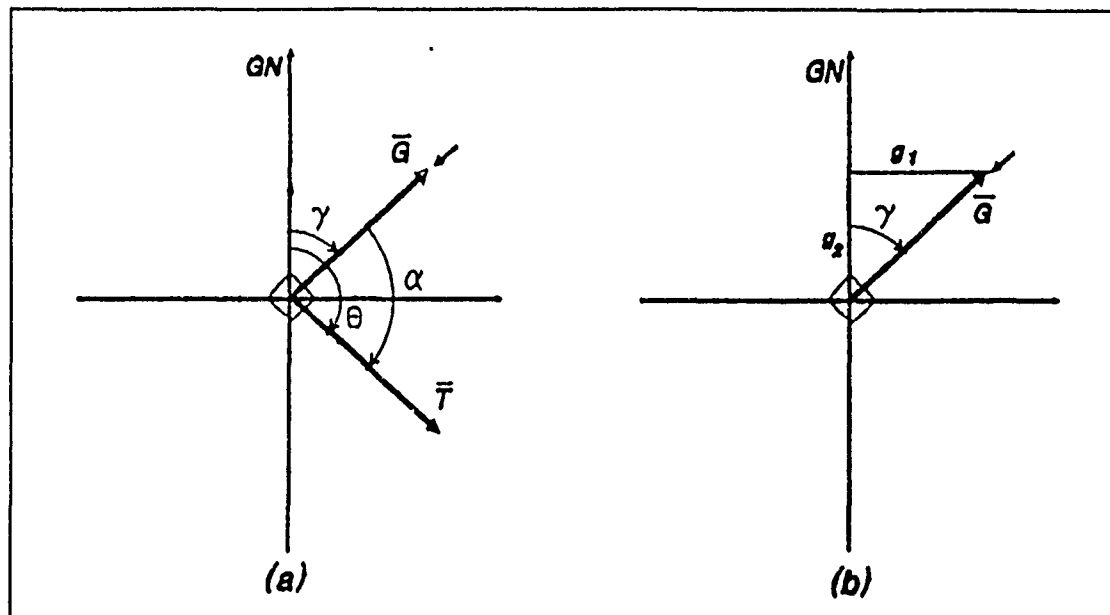


Figure 4. Gun-Target relationship

Missile location and target location are known and given as UTM grid coordinates. Placing the target at the origin,  $\gamma$  may be computed as

$$\gamma = \arctan \frac{g_1}{g_2}.$$

In the case where  $g_2 = 0$ ,  $\gamma = \frac{\pi}{2}$  or  $-\frac{\pi}{2}$ . Note that equation (1) suggests computing the magnitude of each of the vectors to obtain  $\alpha$ . However, since the *dot product* is the sum of the products of the components of the vectors, the numerator may be expressed as

$$\vec{G} \cdot \vec{T} = g_1 t_1 + g_2 t_2.$$

Furthermore,

$$g_1 = \|\vec{G}\| \sin \gamma$$

and

$$g_2 = \|\vec{G}\| \cos \gamma.$$

The components of  $\vec{T}$  follow similarly, so that (1) may be rewritten as

$$\cos \alpha = \frac{\|\vec{G}\| \|\vec{T}\| \sin \gamma \sin \theta + \|\vec{G}\| \|\vec{T}\| \cos \gamma \cos \theta}{\|\vec{G}\| \|\vec{T}\|} \quad (2)$$

Factoring  $\|\vec{G}\| \|\vec{T}\|$ , (2) reduces to

$$\cos \alpha = \sin \gamma \sin \theta + \cos \gamma \cos \theta. \quad (3)$$

Thus, the engagement aspect angle  $\alpha$  is simply

$$\alpha = \arccos (\sin \gamma \sin \theta + \cos \gamma \cos \theta). \quad (4)$$

Equation (4) is the formula which appears in the *AspectAngle* procedure.

Once  $\alpha$  is determined, it is translated to an impact area on the target. Assuming all targets are symmetric about their center of mass, the impact areas are defined as

$$\text{impact area} = \begin{cases} \text{front, for } -45^\circ \leq \alpha \leq 45^\circ \\ \text{flank, for } \begin{cases} 45^\circ < \alpha < 135^\circ \\ 225^\circ < \alpha < 315^\circ \end{cases} \\ \text{rear, for } 135^\circ \leq \alpha \leq 225^\circ \end{cases}$$

Figure 5 depicts the impact areas. The impact area is then passed to the damage assessment procedure to determine the results of the engagement.

#### b. Damage Assessment

Damage assessment is determined by a call to the *AssessDamage* procedure in the *Impact* module. *AssessDamage* requires three input arguments: weapon type, target type, and impact area. The procedure determines the probability of kill for the appropriate missile impacting the target in the given area. A sample is selected from a random number generator and compared to the kill probability. The procedure returns a resulting kill or damage outcome for the engagement.

### 3. Indirect Fires

The indirect fire model in the light infantry attack simulation provides the means of causing attrition to the light force. Assuming that the individual target elements are uniformly distributed throughout the target area, and the incoming rounds impact uniformly throughout the target area, and assuming no rounds land outside the target area and there are no edge effects, let  $P_k$  represent the fraction of target elements killed.

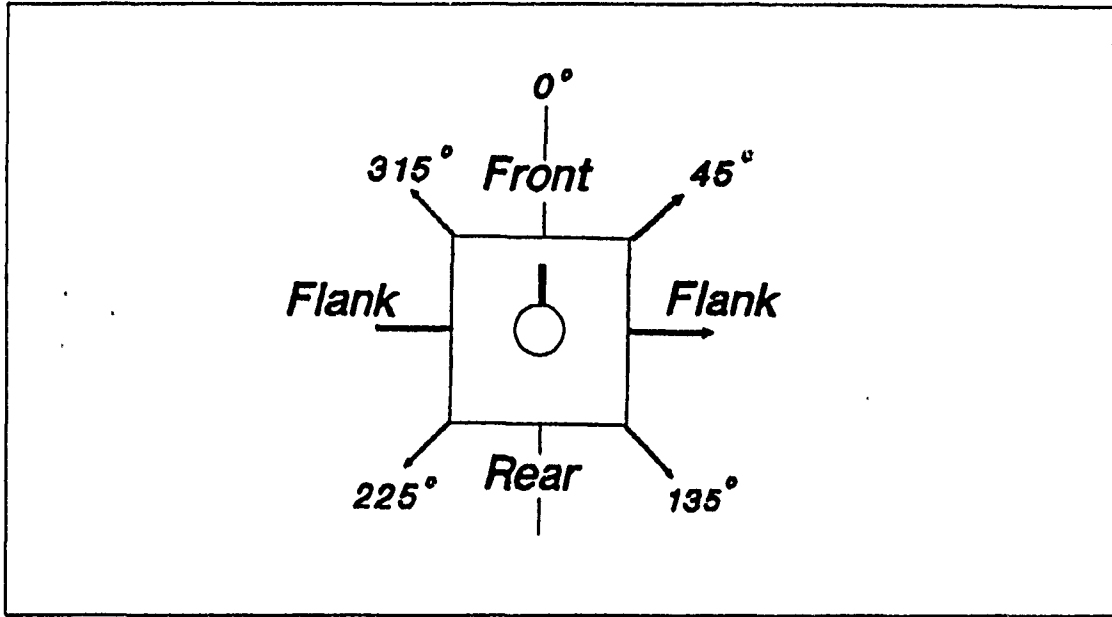


Figure 5. Impact Area Designation

Given these assumptions, the procedure *ScheduleOPFORArtillery* in the *Arty* module employs the confetti approximation

$$P_K = (1 - e^{-\sqrt{z}})^2,$$

where  $z = \frac{na}{A}$ ,  $n$  is the number of rounds fired,  $a$  is the lethal area of one round, and  $A$  is the target area [Ref. 10]. For the purposes of this model, the target area is defined to be a rectangular area measuring 260 meters by 110 meters, which corresponds to the size of the "IFCAS" box used in the NTC rules of engagement [Ref. 11]. The parameter representing the lethal area of one round is an approximation of the lethal area of the OPFOR 122mm high explosive artillery round against infantry troops in the open. Additionally, *ScheduleOPFORArtillery* randomly selects an integer number of rounds per gun, between 1 and 3, fired by an OPFOR battery of six guns. Under this approximation, one scheduled artillery barrage may result in a random casualty assessment ranging from approximately 22% to approximately 45%.

#### 4. Attrition

The light infantry forces modeled in the simulation may be attrited by OPFOR artillery only. In its current configuration, the user schedules the artillery based on an assessment of the probable times at which movement will be compromised or upon



detection of the attack. Any or all of the infantry companies may be scheduled to receive indirect fires. When the simulation time reaches the scheduled time, the company object's *ArtilleryInterrupt* method is invoked. This method accomplishes two tasks: first, it interrupts the unit's current activity, and second, it invokes the platoon-level method *TakeCasualties* and passes the loss percentage.

The *ArtilleryInterrupt* method causes execution of the unit's movement and engagement methods to halt prematurely. A movement interrupt simply causes the unit to elapse additional time while "regrouping" before completing the movement. An engagement interrupt will be passed down to ATGM level and terminate all methods in the engagement sequence. In particular, if the *ATGMObj* is tracking, the missile will be lost; otherwise, the process will wait the constant regroup time before starting over. In addition to early termination of the unit's methods, the unit will be assessed casualties.

Casualties are managed in the model within the platoon's *FPCObj*. Invoking the platoon's *TakeCasualties* method causes the *FPCObj* to implement *DecrementFPC*. The *DecrementFPC* method computes the integer number of casualties represented by the input loss percentage and reduces its strength by the required number. The selection of personnel losses is completely random based on a sample obtained from a random number generator.

## 5. Target Assignments, Reassignments, and Target Handover.

### a. Target Assignments

Assignment of targets to companies is a user provided input presumably based on the assignment and location of company objectives. The data is read in by the *ModelOperationsOverlay* procedure in the *MapRecon* module, and the targets are placed on the company target queue. During execution of the simulation, targets are assigned to platoons upon arrival in the assault position. After the platoons have occupied their respective firing positions, the company invokes *AssignTargets* which assigns targets to platoons according to the following heuristic: start with the most distant target; identify the closest platoon to that target; assign the target to the platoon; continue until all targets have been assigned. This heuristic is one of many alternative methods to optimize the assignment process.

### b. Reassignment

Reassignment of targets occurs within the *ReAssign* method of the *FPCObj*; in other words, targets are reassigned within the platoon. A reassignment occurs when either of two conditions occur: an ATGM system is out of ammunition and its assigned target has not been destroyed, or an ATGM is lost to artillery. Furthermore, should a

condition occur such that the platoon does not have the assets to reassign the target to, the platoon will pass the target back to the company to handover to another platoon.

*c. Target Handover*

The *TargetHandover* method of the *RifleCompanyObj* is called from a subordinate which no longer has the assets to engage a target. A handover can occur within a company; no methodology is provided to pass the target back to the battalion. The handover algorithm first looks at each platoon to identify a candidate. A candidate platoon is one that has ATGM ammunition available and is not currently engaging. The next check identifies a candidate which is currently in range to engage the target and if one exists, immediately assigns the target to the platoon. If, on the other hand, a candidate is not in range, the method then identifies the candidate closest to the target and tells the indicated platoon to move to the appropriate firing position and engage the target. If the company no longer has the assets to engage the target, the target survives and a comment is written to the output file.

## **F. MODEL INPUT**

### **1. Scenario Input**

Scenario data within the light infantry attack simulation are divided into two functional areas: force composition and the light force concept of the operation. The simulation reads scenario input from user developed data files, and dynamically allocates object references at run-time.

*a. Forces*

The model contains all unit related data necessary to allocate the unit objects and set starting force strength. The light force unit object's fields are set within the object's initialization method, while the opposing forces data are input from a data file.

(1) *Friendly Forces.* The light infantry battalion is hierarchically organized with three rifle companies of three rifle platoons each. Each rifle platoon contains an *FPCObj* which contains a numerical representation of each element in the platoon, and an *ATGMObj* for each dragon gunner in the platoon.

Current configuration of the light infantry battalion is based, in part, on the Modification Table of Organization and Equipment (MTOE) for an Infantry Battalion (Airborne). This organization is selected to facilitate the model architecture since the ATGM sections are organic to rifle platoons. The rifle platoon's firepower capability is managed by the *FPCObj*. Starting force strength is set according to the MTOE above, assuming full strength at the start of the battle.

(2) *Opposing Forces.* The OPFOR data is supplied by the user as an input file. A new enemy vehicle object is allocated for each OPFOR system appearing in the data file, and its fields are set with the corresponding data. There is currently no enemy vehicle direct fire capability.

*b. Concept of the Operation*

The light force concept of the operation is input to allow the user to experiment with different tactics and determine simulated outcomes for each approach. The input file is constructed based on the user's map reconnaissance using the "backward planning process". Positional data is input from the objective to the attack position, and lists the unit target assignments. As the data are read in, the positions are stored in a linked-list, and assigned to the appropriate company. Two positions are uniquely identified in the data file: the assault position, and an intermediate objective. Both of these positions have an array of platoon firing positions, such that once the company arrives in that position, the platoons deploy to their respective firing positions. The intermediate objective is employed for the purpose of providing a position such that the attacker can shoot, move, and shoot again.

**2. Model Parameters**

Certain model parameters are fixed at compile-time. These include, for example, the cross country movement rates of dismounted troops at night, or the time required to prepare a Dragon for firing. Where available, the value of the input parameter is obtained from an appropriate field manual (FM). Other model parameters are input at run-time. These parameters include, for example, ATGM kill probabilities and weapon characteristics. Whenever feasible, parameters are read from a data file to provide the user as much flexibility as possible.

**G. MODEL OUTPUT**

The model writes to three output files: the engagement history file, the attrition data file, and the attack output file. During a walk-through, all three files are active. If replicating, only the attack output file is active. The engagement history file contains a detailed listing of each engagement, by system identity and target identity, and the result of the engagement. Also included is the re-assignment and target handover sequence. The attrition file records the losses to each platoon from indirect fires. The attack output file contains the kill data and measure of effectiveness for destruction for the simulation run.

## IV. SIMULATION ANALYSIS

### A. GENERAL

To demonstrate the utility of the model as both an analytic and planning tool, three scenarios are developed. The baseline model replicates an actual NTC deliberate attack mission during a heavy/light rotation; the results of the simulation can be compared to the results achieved on the battlefield and an analysis performed to highlight differences. The two additional scenarios demonstrate the use of the simulation as a planning tool and allow the user to compare results of alternative tactical plans with those of the baseline model. The two alternative plans use the same OPFOR situation as the baseline model. In general terms, the baseline model may be characterized as a frontal attack, while the alternatives represent a rear attack and a flank attack.

### B. OUTPUT ANALYSIS

The light infantry attack simulation is a *terminating simulation* [Ref. 12: p. 280]. The desired measure of performance for the model is defined as the number of enemy vehicles destroyed when the friendly forces are no longer able to engage targets. The simulation terminates and the number of OPFOR kills is reported to the *MOEmean* and *MOEvariance* procedures in the *MOE* modules. These procedures maintain running means and variances over the input number of replications. Let  $X$  be the random variable of interest (the MOE for a single replication), then for fixed sample size  $n$ ,

$$\bar{X}(n) \pm t_{n-1, 1-\alpha/2} \sqrt{\frac{s^2(n)}{n}}$$

yields an approximate  $100(1 - \alpha)$  percent confidence interval, ( $0 < \alpha < 1$ ), for the true mean  $\mu$ , where  $\bar{X}(n)$  is the sample mean and  $s^2(n)$  is the sample variance [Ref. 12: p. 288]. For the purposes of this analysis, sample size  $n = 500$  and significance level  $\alpha = 0.05$ .

### C. THE BASELINE MODEL

The baseline model serves as a point of departure for comparison of alternative tactical plans and outcomes. Operational data for the selected battle is extracted from the numerous media available at the CTC Archive at ARI--POM. Selection of a battle upon which to develop the baseline model was arbitrary; however, numerous battles were screened to ensure conformity with the typical *modus operandi* discussed in Chapter

II, and to select a battle which produced favorable results in terms of the measures of effectiveness. A brief description of the selected battle follows.

### 1. NTC Heavy/Light Mission AA89xxxx

NTC Heavy/Light mission AA89xxxx is a deliberate attack mission of an armored task force with a light infantry battalion. The light infantry battalion conducted a night attack to seize objectives, orient fires towards the enemy to the west and assist the forward passage of the armored task force. In terms of destruction of the OPFOR, the attackers destroyed 66% of the enemy (Chapter II, Table 1, No. 11). However, the attackers also suffered 80% casualties (Chapter II, Table 3, No. 11). Of the enemy vehicles destroyed, one is attributed to a light force Dragon. In terms of infantry casualties, it must be noted that OPFOR direct and indirect fires attrited one infantry company to five personnel, rendered another ineffective, and produced light casualties on the third. Finally, as the heavy task force passed through the infantry positions, it became decisively engaged by OPFOR elements to the west and north not detected by the light force. [Ref. 13]

#### a. *Battle Replay with GNATT II*

GNATT II is the ARI--POM's General-purpose NTC Analysis Training Tool. GNATT II provides a personal computer capability for graphical playback of the NTC data archive. GNATT II programs read four data files which produce representations of units, weapon systems, engagements, and player positions. GNATT II enables the user to portray the battlefield (terrain is not depicted) with individual vehicles emplaced and identified according to data collected by the NTC's instrumentation system and position location devices. [Ref. 14]

The utility of GNATT II, in the context of the light infantry attack simulation, is that the user identifies the OPFOR vehicles by type, and extracts the actual enemy positions from an NTC battle. This data is entered into the OPFOR data file and read in by the *ModelEnemyDefense* procedure, so that the light infantry attack simulation better approximates the actual battlefield conditions. The baseline model represents the enemy situation as indicated by the GNATT II display screen shown in Figure 6 (only OPFOR vehicles shown). The remaining elements of the baseline model scenario follow from extracts of the NTC unit take-home package and operations overlays.

### 2. Scenario Input

#### a. *Scheduling of Indirect Fires*

Indirect fires are one of the leading causes of infantry attrition at the NTC. The light infantry attack simulation provides a means of reducing infantry effectiveness

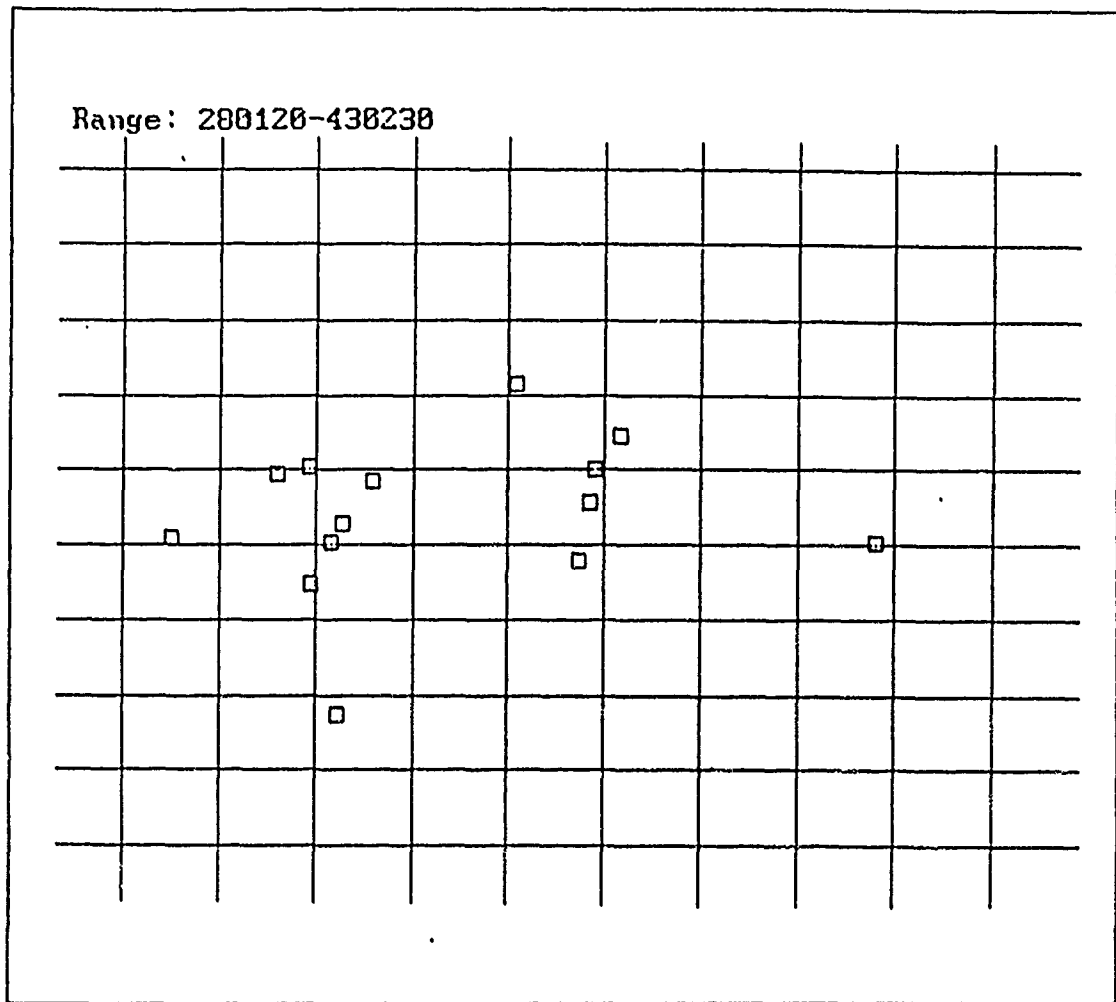


Figure 6. GNATT II Replay of AA89xxxx

by reducing their strength with scheduled artillery effects. As indicated, indirect fires rendered two rifle companies ineffective during AA89xxxx. To ensure consistency between the three modeled scenarios, indirect fires are scheduled against two companies during movement to the assault position, and against the third company while it is in the assault position preparing to engage targets.

In terms of light force losses, the mean level of light force attrition for 500 iterations is 33.9972%. This may be interpreted in terms of the Survival MOE as approximately 66%; however, there is insufficient data to compare with losses at the NTC. A sample attrition output file is contained in Appendix D.

### *b. Forces*

(1) *Opposing Forces.* The GNATT II display screen in Figure 6 indicates the positions of OPFOR vehicles during AA89xxxx. In particular, there are 14 OPFOR vehicles in this battle, consisting of 11 BMPs, one T72, one BRDM, and one ZSU 23-4. In addition to the identification and location information, the user also inputs the target orientation. Target orientation is simply the user's estimate of the principal field of view for each target. Appendix B contains the identity, location, and orientation for each OPFOR vehicle from AA89xxxx. The opposing forces scenario is identical for each of the baseline model and the two alternative models.

(2) *Friendly Force Concept of the Operation.* The scenario input for friendly forces is read in by the *ModelOperationsOverlay* procedure in the *MapRecon* module. The light infantry battalion represented in the light infantry attack simulation starts the mission at full strength, consisting of three rifle companies of three platoons each. Platoon starting strength includes two ATGM gunners with two missiles each. Friendly force structure is identical for all three tactical alternatives. Movement routes for the rifle companies are input to mirror the original operations overlay for mission AA89xxxx, and objectives are assigned with corresponding targets for each objective area. The objectives differ from the original graphics only so that the unit ATGMs will be within range of the OPFOR positions indicated in the GNATT II display. The baseline model concept of the operation is depicted in Figure 7.

### **3. Baseline Model Results**

As indicated previously, the baseline model can be characterized as a frontal attack. The light forces begin the battle with 18 ATGM gunners, or 36 missiles with which to engage the enemy vehicles. In a trial run of the model without OPFOR artillery play, the light force successfully destroyed nine vehicles. A sample engagement history is contained at Appendix E. Returning to the Destroy MOE developed in Chapter II,

$$\text{Destroy MOE} = \frac{\text{Total OPFOR Destroyed}}{\text{Total OPFOR Starting}},$$

the resulting effectiveness of an undetected, unattritted light force is approximately 64%. Although this level of effectiveness is unrealistic given the environment, it serves as a reference point, within the model architecture, to compare levels of effectiveness under less than ideal conditions. Subsequently, a run of the model *with* an artillery strike

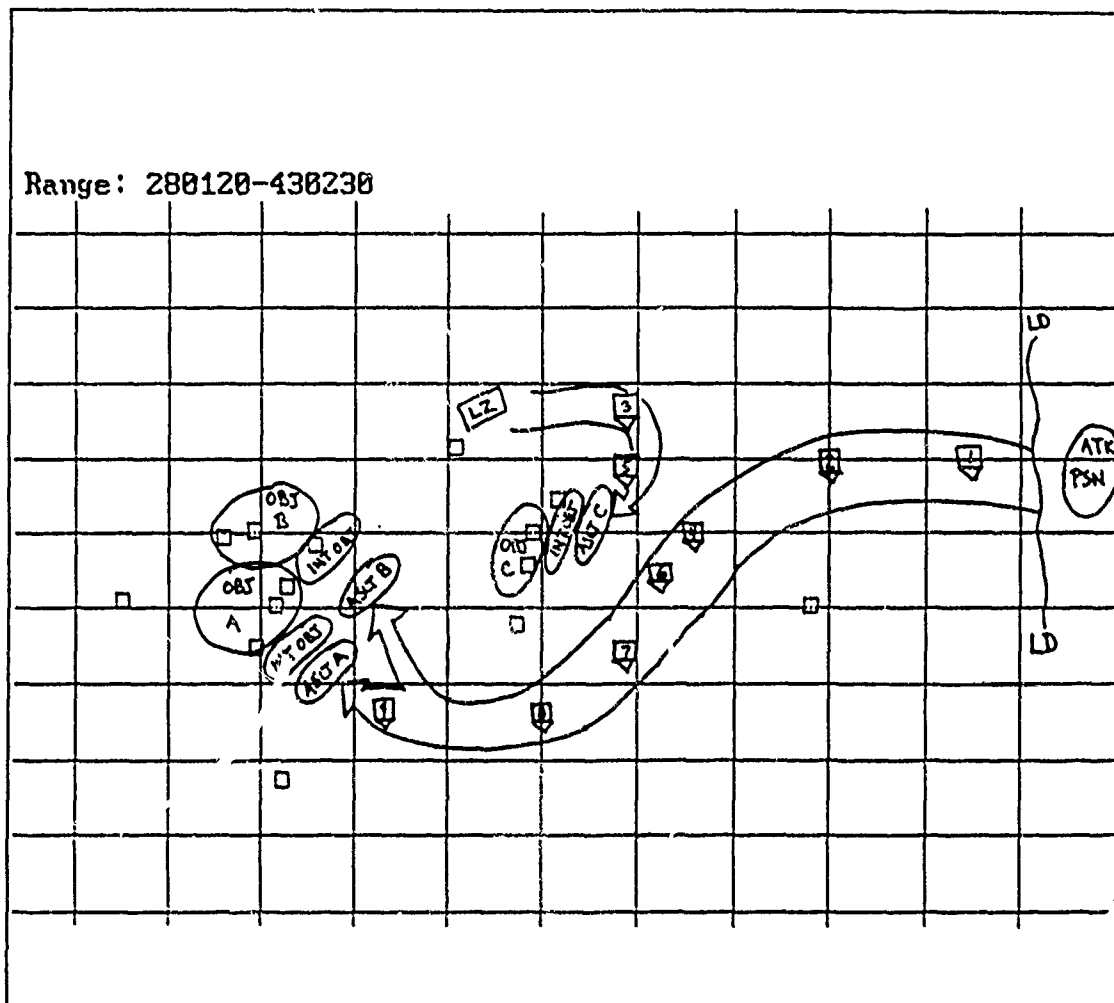


Figure 7. Baseline Model Operations Overlay

directed at each company produced a Destroy MOE of approximately 57%, or eight OPFOR vehicles destroyed. Furthermore, replicating the model through 500 iterations, the resulting mean Destroy MOE was 58.81%, with a variance of 0.0105, so that a 95% confidence interval on the mean destruction of OPFOR in this attack scenario is

$$0.5791 \leq \mu \leq 0.5971.$$

Appendix F contains results of the baseline model, including the statistical summary for the replications and attack output files for both events with and without artillery.



## **D. A REAR ATTACK PLAN**

### **1. Concept of the Operation**

The rear attack plan assumes insertion of the force to a landing zone behind enemy lines. Movement routes position the friendly forces to the rear of the enemy prior to engagement. Companies retain objective and target assignments similar to the baseline model. The rear attack concept of the operation is depicted in Figure 8.

### **2. Model Results**

In a trial run of the model without OPFOR artillery, the light force successfully destroyed nine vehicles, or a Destroy MOE for an unattritted force of 64.29%. A trial run of the model with an artillery strike directed at each company produced a Destroy MOE of approximately 57.14%, or eight OPFOR vehicles destroyed. Replicating the model through 500 iterations, the resulting mean Destroy MOE is 67.57%, with a 1.58% variance. The resulting confidence interval on the mean destruction of OPFOR in this attack scenario is

$$0.6647 \leq \mu \leq 0.6867.$$

Results of the rear attack model are contained in Appendix G.

## **E. A FLANK ATTACK PLAN**

### **1. Concept of the Operation**

The flank attack plan assumes insertion of the force to a landing zone to the north of the enemy's positions. Movement routes position the forces on the northern flank of the enemy prior to engagement. The flank attack concept of the operation is depicted in Figure 9.

### **2. Model Results**

In a trial run of the model without artillery, the light force successfully destroyed 11 vehicles, a measure of effectiveness for an unattritted force of 78.57%. A trial run of the model with artillery produced a Destroy MOE of approximately 50%, or seven OPFOR vehicles destroyed. Replicating the model through 500 iterations, the resulting mean Destroy MOE was 63.30%, with a 1.96% variance, producing a confidence interval on the mean destruction of OPFOR in this attack scenario of

$$0.6207 \leq \mu \leq 0.6453.$$

Results of the flank attack model are contained in Appendix H.

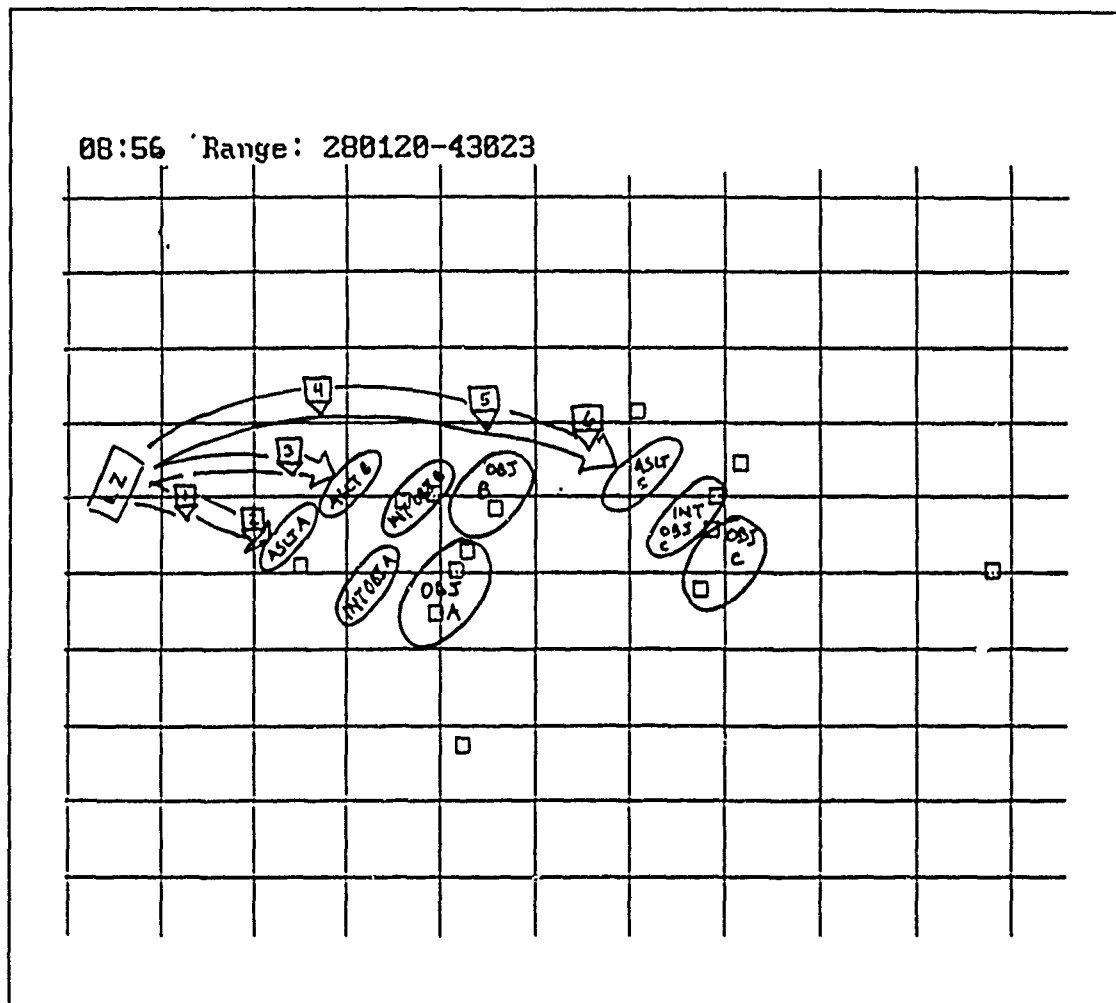


Figure 8. Rear Attack Operations Overlay

## F. COMPARISON OF RESULTS

The results of the simulation experiment support the tactical assertion that it is to the attacker's advantage to approach the objective from a direction the enemy is not expecting. An advantage in this tactic is that it exposes an enemy weakness to the effects of friendly direct fire weapon systems. In particular, it is generally the case that armored vehicles are less susceptible to weapons effects when struck from the front as opposed to the flank or rear. Typically, armor protection is increased on the frontal slopes of these vehicles, and the target silhouette, when viewed from the front, is minimized. Therefore, it is usually to the attacker's advantage to infiltrate to a position to

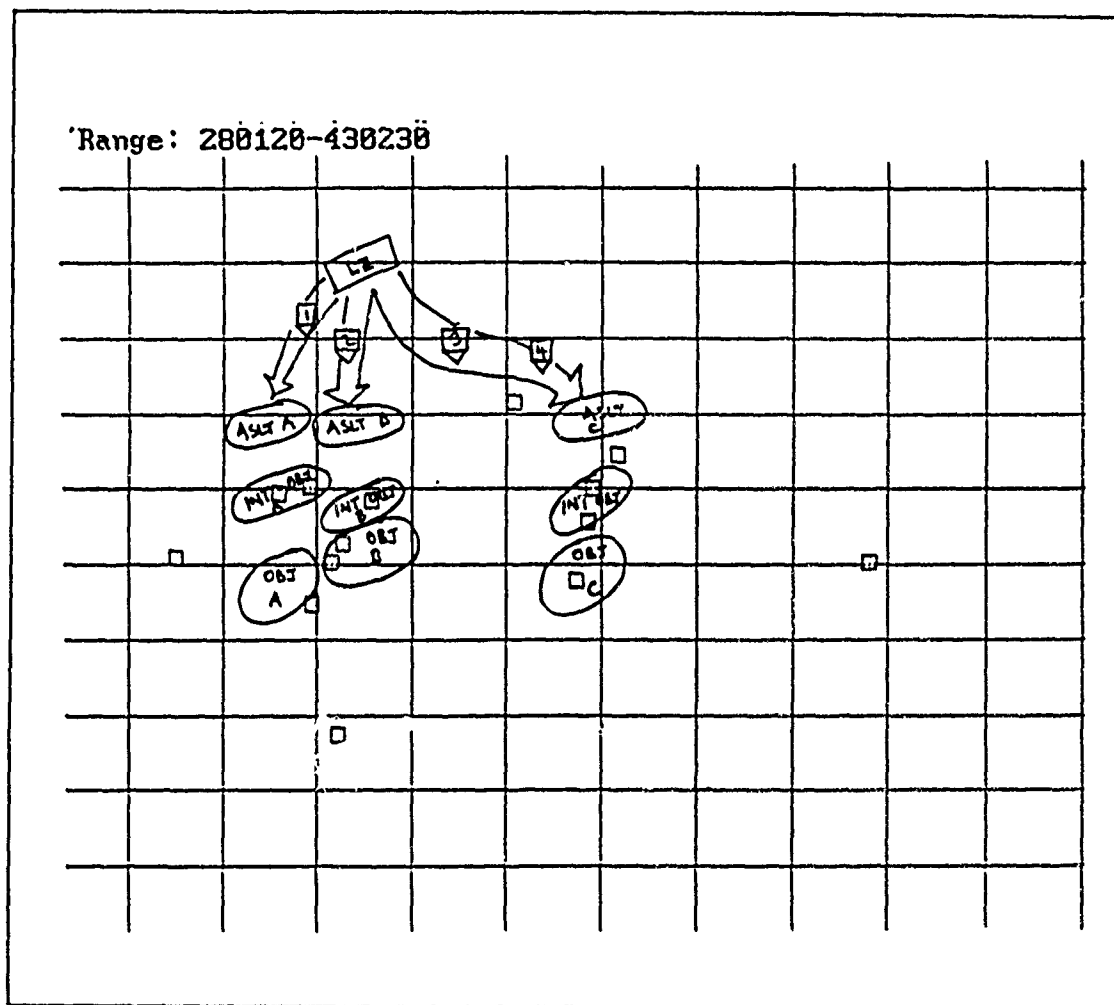


Figure 9. Flank Attack Operations Overlay

the rear or the flank of the enemy to maximize the probability of hit and probability of kill.

An initial comparison of the results of each scenario indicates a significant difference in the expected measure of effectiveness for frontal, flank, and rear attacks. This result is not offered as evidence to claim the superiority of one tactic over the other; however, it follows the intuition that, under similar conditions, units might be expected to achieve more destructive effect on enemy forces while attacking from the flank or rear, as opposed to a frontal attack. Furthermore, this result tends to verify the utility of the model as a planning and analytic tool. Figure 10 depicts the range of the confidence intervals for each of the scenarios. This plot clearly indicates a difference between the scenarios.

However, Figure 11 depicts the probability density for the sample results of each scenario for 500 replications. Due to the amount of variation in the distributions of the results of each scenario, further analysis to determine whether one scenario is statistically significantly different from another will reinforce these general conclusions.

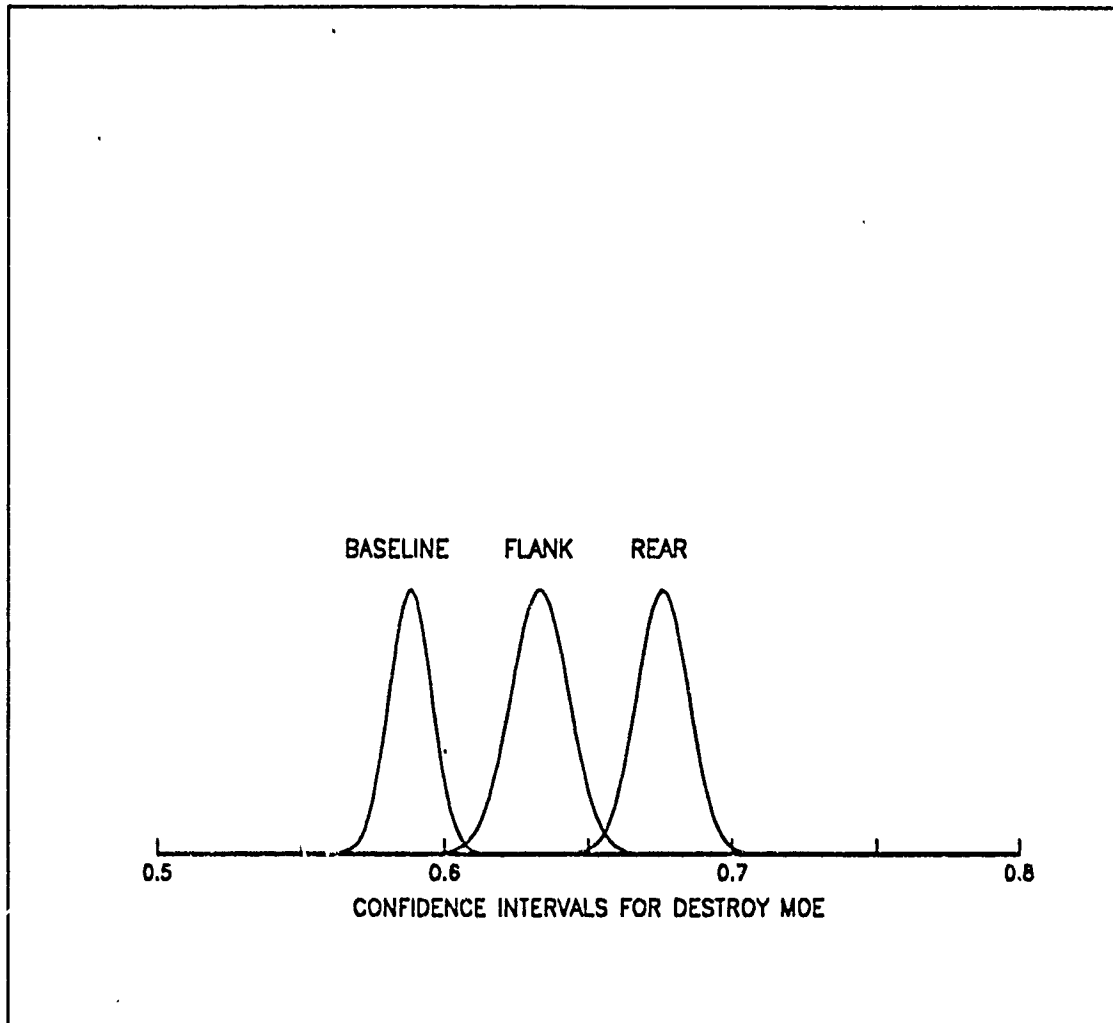


Figure 10. Confidence Intervals for Each Scenario

It is possible to reduce the variance of an output random variable without disturbing its expected value, thus yielding greater precision, i.e., smaller confidence intervals [Ref. 12: p. 349]. The method of *common random numbers* (CRN), is a variance reduction technique applied to measure the relative performance of the model under the three scenarios. Since each scenario is run under identical conditions, and calls to the random number generators produce synchronized streams of random numbers, it is desired to

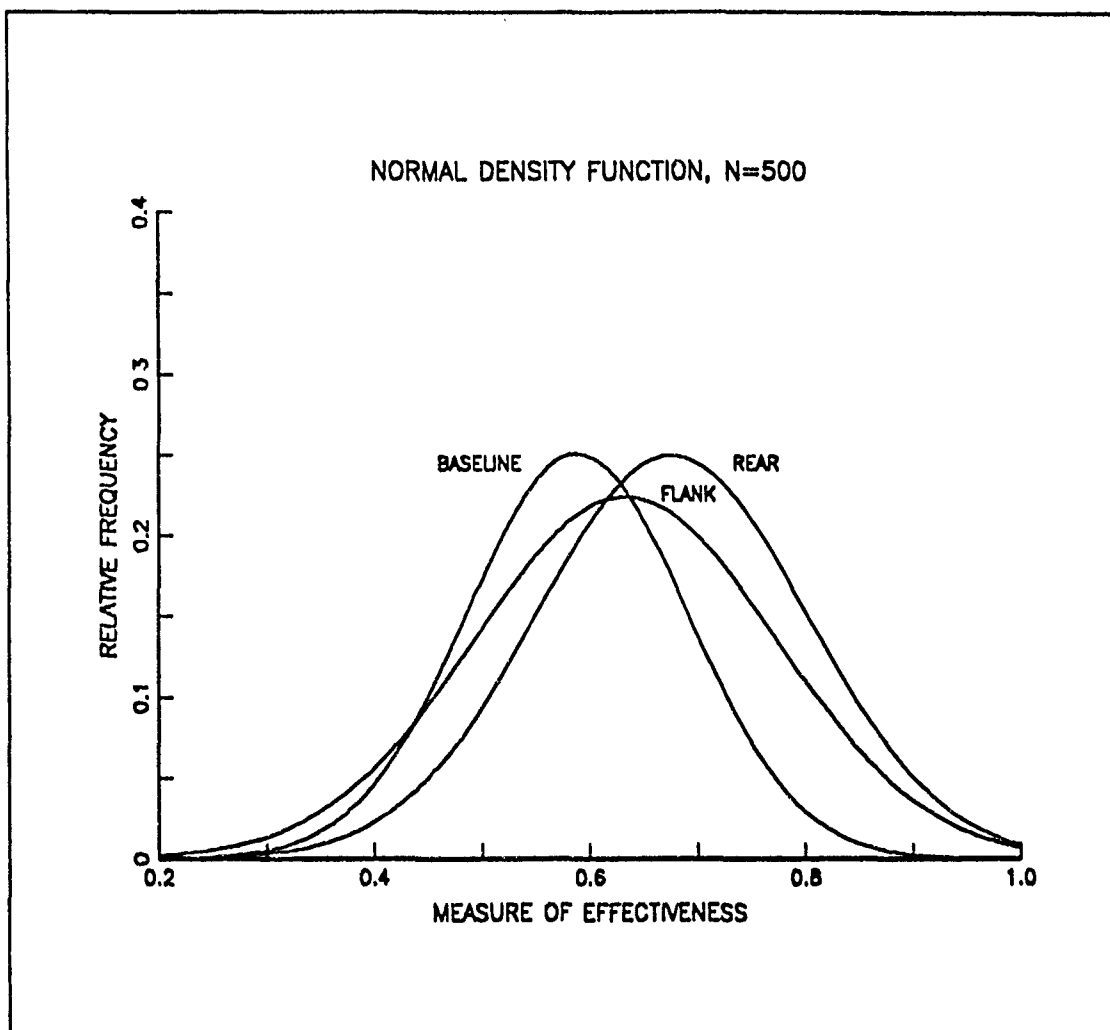


Figure 11. Scenario Distributions

estimate the difference between the expected values of each scenario run,  $\zeta$ , and produce a confidence interval on this result. The output variables, or replication MOEs,  $X_{Bj}$ ,  $X_{Rj}$ , and  $X_{Fj}$ , where  $B$ ,  $R$ , and  $F$  represent the baseline, rear, and flank attack scenarios respectively on the  $j$ th independent replication, are correlated random variables. By the method of common random numbers, letting  $Z_j = X_{Rj} - X_{Bj}$  for  $j = 1, 2, \dots, n$ , then  $\bar{Z}(n) = \sum_{j=1}^n Z_j / n$  is an unbiased estimator of  $\zeta = E(X_{Rj}) - E(X_{Bj})$ . Since the  $Z_j$ 's are IID random variables,

$$\begin{aligned} \text{Var}(\bar{Z}(n)) &= \frac{\text{Var}(Z_j)}{n} \\ &= \frac{\text{Var}(X_{Rj}) + \text{Var}(X_{Bj}) - 2\text{Cov}(X_{Rj}, X_{Bj})}{n}, \end{aligned}$$

so that any positive correlation between  $X_{Rj}$  and  $X_{Bj}$ , has  $\text{Cov}(X_{Rj}, X_{Bj}) > 0$ . Consequently, variance of  $\bar{Z}(n)$  is reduced. [Ref. 12: p. 351]. Furthermore, the form of the confidence interval is

$$\bar{Z}(n) \pm t_{n-1, 1-\alpha/2} \sqrt{s_Z^2/n},$$

where  $s_Z^2$  is the variance of the  $Z_j$ 's [Ref. 8: p. 49].

The method of common random numbers is applied to each scenario, with results shown in Table 5. In each case, 0 is not contained in the confidence interval, so it may be concluded that there is significant difference between results of the three scenarios. Interestingly, CRN reduced the total variance in the Rear-Baseline samples by 0.0066, or approximately 25.2%, reduced the variance in the Flank-Baseline samples by 0.0079, or 26.4%, and reduced the variance in the Rear-Flank samples by 0.0084, or 23.9%. Numerous factors contribute to these results, notably the specific input parameters for probability of kill. However, model *validation*, and the associated sensitivity analysis, is beyond the scope of this thesis.

**Table 5. RESULTS OF CRN TEST OF DIFFERENCE**

Test	Mean Performance	Standard Deviation	Confidence Interval (95%)	
			Lower Limit	Upper Limit
$X_{RJ} - X_{BJ}$ (Rear-Baseline)	0.08756	0.14024	0.07527	0.09985
$X_{FJ} - X_{BJ}$ (Flank-Baseline)	0.04484	0.14878	0.03181	0.05788
$X_{RJ} - X_{FJ}$ (Rear-Flank)	0.04271	0.16417	0.02833	0.05711

## **V. CONCLUSIONS AND RECOMMENDATIONS**

### **A. CONCLUSIONS**

The purpose of the simulation study is to produce a modeling tool to experiment with and to analyze light infantry operations in a mid-to-high intensity environment. The NTC's training environment and data collection capacity provide background information. The initial data collection and analysis suggest almost negligible light force contribution to overall mission effectiveness. There are several factors which contribute to light force effectiveness in this environment. The simulation model, then, provides a tool to analyze both: "What results might we have been able to achieve?", and "What results might we have achieved if we had attacked this way?"

The results from three different tactical experiments produced distinct measures of effectiveness, as measured in terms of OPFOR destruction. The results follow intuitive lines: flank and rear attacks would generally be expected to produce better results than a frontal attack. Because the model compares random numbers to input parameters, obviously the more accurate the input parameters, the more accurately the simulation results should compare with expected battlefield results. The model can be readily adapted to read such data.

Using approximations of the effectiveness of the Dragon missile system against various OPFOR vehicles, the simulation results of the baseline model suggested that light infantry units operating at night should be able to achieve significantly better results than are obtainable at the CTCs. One possible explanation is the lack of a compatible night firing MILES device for the Dragon.

As an initial modeling effort, this model represents a detailed simulation of the events on the battlefield, from movement along prescribed routes, to assignments and engagements of targets according to steps commonly used in training. This model, more than anything else, represents a low-cost, highly exportable planning and analysis alternative to large scale combat models in use today. Its modular development allows adaptation to other models, and more importantly, allows growth and follow-on development to expand its utility.

### **B. RECOMMENDATIONS**

One of the early assertions made in this research is the inability of our training centers to provide an environment which facilitates employing forces against an enemy

they are capable of defeating. Clearly, light infantry is an effective force in an environment such as the NTC; however, the continued employment of light infantry against enemy armored and mechanized forces, in other than close terrain, is doctrinally untenable. Doctrinal complementary force operations must stress the notion of employing light forces in operations against enemy battlefield operating systems, other than his maneuver forces, to maximize their effectiveness and create a dilemma for the enemy.

This model has several limitations, principally the lack of OPFOR direct fires. Continued development to improve such shortcomings will improve the results of the model in general, and more specifically, as a valuable tool for planning and analyzing complementary force operations. The scenarios developed to analyze employment of light forces in this research also consist entirely of operations in which the light force is attacking the enemy's heavy maneuver forces. However, further scenario development to portray OPFOR CS and CSS elements throughout the depth of the battlefield is entirely possible and may demonstrate the utility of the model in exploring employment of light forces against targets other than heavy maneuver forces. Furthermore, in the context of heavy/light operations, the development of a complementary heavy force attack simulation would greatly improve this model's utility. The results of the light force operations establish the input parameters for the heavy model, so that a more accurate picture of heavy/light effectiveness may be obtained.



## APPENDIX A. MODSIM CODE

### A. ATTACK

MAIN MODULE Attack;

```
FROM SimMod    IMPORT StartSimulation, SimTime, ResetSimTime;
FROM CRTMod    IMPORT ClearScreen;
FROM Unit      IMPORT BattalionObj;
FROM MapRecon  IMPORT ModelOperationsOverlay;
FROM OPFOR     IMPORT ModelEnemyDefense;
FROM Arty      IMPORT ScheduleOPFORArty, Pk;
FROM Weapons   IMPORT ReadMissileData;
FROM Globals   IMPORT Setup, UnitNameType;
FROM Menu      IMPORT RunMenu1, numberOfReplications,
                    replicating, walkingThru, CleanUp;
FROM MOE       IMPORT Mean, MOEmean, ReportStats,
                    meanMissionTime, percentAttrition,
                    meanAttritionForThisRun, TotalOPFORlosses;
```

VAR

```
  i, j          : INTEGER;
  LightFighters : BattalionObj;
```

BEGIN

```
  ClearScreen;
  RunMenu1;
  Setup;
  ReadMissileData;
```

```
  FOR i := 0 TO numberOfReplications - 1
    meanAttritionForThisRun := 0.0;
```

```
    ModelEnemyDefense;
    ModelOperationsOverlay;
    ScheduleOPFORArty;
    ResetSimTime(0.0);
    NEW(LightFighters);
    IF walkingThru
      OUTPUT("The battalion is executing the mission.");
    END IF;
    TELL LightFighters TO ExecuteMission;
```

```
  StartSimulation;
```

```
  MOEmean(i, FLOAT(TotalOPFORlosses));
  meanMissionTime := Mean(i, meanMissionTime, SimTime()/3600.0);
  FOR j := 0 TO 2
    meanAttritionForThisRun := Mean(j, meanAttritionForThisRun,
                                     Pk[VAL(UnitNameType,j)]);
```

```
  END FOR;
```

```

percentAttrition := Mean(i, percentAttrition,
                          meanAttritionForThisRun);

DISPOSE(LightFighters);
DISPOSE(Pk);
CleanUp;
IF replicating
    OUTPUT("Run number ",i+1," complete.");
END IF;
END FOR;

ReportStats;
OUTPUT("MISSION ACCOMPLISHED");
IF walkingThru
    OUTPUT("Ended normally at:");
    OUTPUT("H + ", SimTime()/3600.0," hrs.");
    OUTPUT;
END IF;
OUTPUT("Look in file attack.out for results of the battle.");

END MODULE.

```

## B. GLOBALS

DEFINITION MODULE Globals;

FROM IOMod IMPORT StreamObj;  
FROM RandMod IMPORT RandomObj;

TYPE

UnitNameType = (A, B, C, D);

WeaponsStatusType = (HOLD, TIGHT, FREE);

TargetStatusType = (missed, damaged, killed);

TransType = (Foot, Truck, AirAssault);

VisCondType = (Day, Night); (\* Visibility Condition \*)

MovementRateList = ARRAY INTEGER, INTEGER OF REAL;  
(\* ARRAY TransType, VisCondType OF REAL; \*)

ConversionFactorList = ARRAY TransType OF REAL;  
(\* to convert movement rates to m/sec \*)

PROCEDURE Setup;

PROCEDURE ReadTransportationData;

VAR

OutputFile,  
EngagementHistory,  
AttritionFile : StreamObj;  
MovementRate : MovementRateList;  
CF : ConversionFactorList;  
WeaponsStatus : WeaponsStatusType;  
RegroupTime : REAL;  
VisCond : VisCondType;  
BDA,  
HitOrMiss,  
RandomCasualty,  
RoundGenerator : RandomObj;

END MODULE.

IMPLEMENTATION MODULE Globals;

FROM IOMod   IMPORT StreamObj, FileUseType(Input, Output);  
FROM RandMod  IMPORT FetchSeed;  
FROM Debug   IMPORT TraceStream;

PROCEDURE Setup;

BEGIN

    WeaponsStatus := HOLD;  
    VisCond   := Night;  
    RegroupTime := 150.0; (\* 2 and a half minutes to regroup \*)  
    NEW(OutputFile);  
    ASK OutputFile TO Open("attack.out", Output);  
    NEW(EngagementHistory);  
    ASK EngagementHistory TO Open("engage.hst", Output);  
    NEW(AttritionFile);  
    ASK AttritionFile TO Open("attrit.out", Output);  
    NEW(BDA);  
    ASK BDA TO SetSeed(FetchSeed(1));  
    NEW(HitOrMiss);  
    ASK HitOrMiss TO SetSeed(FetchSeed(2));  
    NEW(RandomCasualty);  
    ASK RandomCasualty TO SetSeed(FetchSeed(3));  
    NEW(RoundGenerator);  
    ASK RoundGenerator TO SetSeed(FetchSeed(4));  
    NEW(TraceStream);  
    ASK TraceStream TO Open("trace.out", Output);  
    ASK TraceStream TO TraceOff;  
    ReadTransportationData;

END PROCEDURE;  (\* Setup \*)

(\* ----- \*)

PROCEDURE ReadTransportationData;

VAR

    i                         : INTEGER;  
    TransportationDataFile   : StreamObj;  
    nilentry                 : STRING;

BEGIN

    i := 0;  
    NEW(TransportationDataFile);  
    ASK TransportationDataFile TO Open("trans.dat", Input);  
    NEW(MovementRate, ORD(Foot)..ORD(AirAssault), ORD(Day)..ORD(Night));  
    NEW(CF, Foot..AirAssault);  
    ASK TransportationDataFile TO ReadLine(nilentry);  
    WHILE NOT ASK TransportationDataFile eof  
        ASK TransportationDataFile TO ReadString(nilentry);  
        ASK TransportationDataFile TO  
            ReadReal(MovementRate[i,i]);  
        ASK TransportationDataFile TO  
            ReadReal(MovementRate[i,i+1]);  
        ASK TransportationDataFile TO  
            ReadReal(CF[VAL(TransType,i)]);  
    INC(i);

```
END WHILE;  
ASK TransportationDataFile TO Close;  
DISPOSE(TransportationDataFile);  
  
END PROCEDURE; (* ReadTransportationData *)  
  
END MODULE.
```

## C. UNIT

DEFINITION MODULE Unit;

```
FROM SimMod      IMPORT TriggerObj;
FROM GrpMod       IMPORT StackObj;
FROM MapRecon     IMPORT PositionRecordType;
FROM OPFOR        IMPORT EnemyVehicleObj;
FROM Globals      IMPORT WeaponsStatusType, TransType, UnitNameType;
```

TYPE

```
UnitObj = OBJECT (* generic unit object *)
  myHQ           : UnitObj;
  identity       : STRING;
  location       : PositionRecordType;
  myFirePower    : ANYOBJ;
  moving,
  set,
  outOfATGMammo,
  engaging,
  engagementComplete,
  finalAssault   : BOOLEAN;
  mvtStartTime,
  movementTime   : REAL;
  ASK METHOD UpdateStatus;
  ASK METHOD SetLocation(IN position : PositionRecordType);
  TELL METHOD TargetHandover(IN target : EnemyVehicleObj;
                             IN firingPosition : STRING);
  TELL METHOD MoveTo (IN position : PositionRecordType;
                     IN method   : TransType);
```

END OBJECT;

```
RiflePlatoonObj = OBJECT (UnitObj)
  ASK METHOD PltInit(IN HQ : UnitObj;
                    IN id : STRING);
  ASK METHOD TakeCasualties(IN lossPercentage: REAL);
  TELL METHOD OccupyFiringPosition(IN firingPosition : STRING);
  ASK METHOD PrepareToEngage(IN pltTargetList : StackObj);
  TELL METHOD Engage;
  TELL METHOD InterruptEngage;
  TELL METHOD FinalAssault;
END OBJECT;
```

PlatoonList = ARRAY INTEGER OF RiflePlatoonObj;

```
RifleCompanyObj = OBJECT(UnitObj)
  unitName       : UnitNameType;
  platoon        : PlatoonList;
  targetList     : StackObj;
  alreadyFired   : BOOLEAN;
  movementComplete : TriggerObj;
  ASK METHOD CompanyInit(IN HQ : UnitObj;
```

```

                                IN Name : UnitNameType);
TELL METHOD ExecuteMovementPlan;
TELL METHOD ArtilleryInterrupt(IN casualtyAssessment : REAL);
TELL METHOD AssignTargets;
TELL METHOD Hold(IN target : EnemyVehicleObj;
                IN firingPosition : STRING );
TELL METHOD Attack;
TELL METHOD FinalAssault;
OVERRIDE
    TELL METHOD TargetHandover(IN target : EnemyVehicleObj;
                              IN firingPosition : STRING);
    ASK METHOD UpdateStatus;
END OBJECT;

CompanyList = ARRAY UnitNameType OF RifleCompanyObj;

BattalionObj = OBJECT(UnitObj)
    company    : CompanyList;
    execute    : TriggerObj;
    ASK METHOD ObjInit;
    TELL METHOD ExecuteMission;
    OVERRIDE
        ASK METHOD UpdateStatus;
END OBJECT;

VAR
    i          : INTEGER;
    name       : UnitNameType;
    firstTimeSet : BOOLEAN;

END MODULE.

```

IMPLEMENTATION MODULE Unit;

```
FROM SimMod      IMPORT SimTime, TriggerObj, Interrupt;
FROM UtilMod     IMPORT Delay, MicroDelay;
FROM CRTMod      IMPORT ClearScreen;
FROM GrpMod      IMPORT StackObj;
FROM MathMod     IMPORT CEIL;
FROM FPC         IMPORT FPCObj;
FROM OPFOR       IMPORT EnemyVehicleObj;
FROM Arty        IMPORT Pk, ImpactTimeA, ImpactTimeB, ImpactTimeC;
FROM Menu        IMPORT walkingThru, playingArty;
FROM Globals     IMPORT ALL TransType, VisCond, MovementRate, CF,
                      ALL UnitNameType, RegroupTime, OutputFile,
                      EngagementHistory, WeaponsStatus,
                      ALL WeaponsStatusType;
FROM MapRecon    IMPORT Distance, PositionRecordType, ALL SymbolType,
                      UnitTargetList, UnitRoute;
```

```
(* ***** *)
OBJECT UnitObj;
(* ***** *)
```

ASK METHOD SetLocation(IN position : PositionRecordType);

VAR

fpc : FPCObj;

BEGIN

fpc := myFirePower;

location := CLONE(position);

ASK fpc TO SetLocation(location.coordinate);

END METHOD; (\* SetLocation \*)

```
(* ----- *)
```

ASK METHOD UpdateStatus;

VAR

fpc : FPCObj;

BEGIN

fpc := myFirePower;

IF NOT finalAssault

set := ASK fpc ready;

engagementComplete := ASK fpc firingComplete;

IF engagementComplete OR set

ASK myHQ TO UpdateStatus;

END IF;

END IF;

engaging := ASK fpc engaging;

outOfATGMammo := ASK fpc outOfATGMammo;

END METHOD; (\* Platoon UpdateStatus \*)

```
(* ----- *)
```

TELL METHOD TargetHandover(IN target : EnemyVehicleObj;

IN firingPosition : STRING);

BEGIN



```

    TELL myHQ TO TargetHandover(target, firingPosition);
END METHOD;

(* ----- *)

TELL METHOD MoveTo (IN position : PositionRecordType;
                  IN method   : TransType);
VAR
    distance, mvtRate : REAL;
    remMvtTime       : REAL;

BEGIN
    moving      := TRUE;
    mvtStartTime := SimTime();
    distance    := Distance(location.coordinate, position.coordinate);
    mvtRate     := CF[method] * MovementRate[ORD(method),ORD(VisCond)];
    movementTime := distance/mvtRate;
    WAIT DURATION movementTime
        DISPOSE(location);
        location := position;
        moving   := FALSE;
    ON INTERRUPT (* determine remaining movement time *)
        remMvtTime := movementTime - (SimTime() - mvtStartTime);
        WAIT DURATION RegroupTime + remMvtTime
            DISPOSE(location);
            location := position;
            moving   := FALSE;
        END WAIT;
    END WAIT;
END METHOD;      (* MoveTo *)

END OBJECT;    (* UnitObj *)

(* ***** *)
OBJECT RiflePlatoonObj;
(* ***** *)

ASK METHOD PltInit(IN HQ : UnitObj;
                 IN id : STRING);
VAR
    fpc : FPCObj;

BEGIN
    myHQ      := HQ;
    identity  := id;
    location  := CLONE(ASK myHQ location);
    NEW(fpc);
    ASK fpc TO FPCInit(SELF);
    myFirePower := fpc;
END METHOD;    (* PltInit *)

(* ----- *)

ASK METHOD TakeCasualties(IN lossPercentage : REAL);
VAR
    fpc : FPCObj;

```

```

BEGIN
    fpc := myFirePower;
    ASK fpc TO DecrementFPC(lossPercentage);
END METHOD;      (* TakeCasualties *)

(* ----- *)

TELL METHOD OccupyFiringPosition(IN firingPosition : STRING);
VAR
    fpc      : FPCObj;
    mvtTime : REAL;

BEGIN
    engaging := TRUE;
    fpc := myFirePower;
    IF location.coordinate = firingPosition
        mvtTime := 0.0;
    ELSE
        mvtTime := Distance(location.coordinate, firingPosition) /
                     (CF[Foot] * MovementRate[ORD(Foot), ORD(VisCond)]);
    END IF;
    WAIT DURATION mvtTime
        location.coordinate := firingPosition;
    END WAIT;
    ASK fpc TO SetLocation(firingPosition);
END METHOD;      (* OccupyFiringPosition *)

(* ----- *)

ASK METHOD PrepareToEngage(IN pltTargetList : StackObj);
VAR
    fpc : FPCObj;

BEGIN
    engaging := TRUE;
    fpc := myFirePower;
    TELL fpc TO PrepareToFire(pltTargetList);
END METHOD;      (* PrepareToEngage *)

(* ----- *)

TELL METHOD Engage;
VAR
    fpc : FPCObj;

BEGIN
    fpc := myFirePower;
    TELL fpc TO Fire;
END METHOD;      (* Engage *)

(* ----- *)

TELL METHOD InterruptEngage;
VAR
    fpc : FPCObj;

```

```

BEGIN
    fpc := myFirePower;
    TELL fpc TO InterruptFire;
END METHOD; (* InterruptEngage *)

(* ----- *)

TELL METHOD FinalAssault;
VAR
    fpc : FPCObj;

BEGIN
    fpc := myFirePower;
    finalAssault := TRUE;
    TELL fpc TO FinalAssault;
END METHOD; (* FinalAssault *)

END OBJECT; (* RiflePlatoonObj *)

(* ***** *)
OBJECT RifleCompanyObj;
(* ***** *)

ASK METHOD CompanyInit(IN HQ : UnitObj;
                      IN Name : UnitNameType);
VAR
    plt : RiflePlatoonObj;
    pltID : STRING;

BEGIN
    CASE Name
        WHEN A : pltID := "A0";
        WHEN B : pltID := "B0";
        WHEN C : pltID := "C0";
    END CASE;
    unitName := Name;
    myHQ := HQ;
    location := UnitRoute[unitName];
    alreadyFired := FALSE;
    targetList := UnitTargetList[unitName];
    NEW(movementComplete); (* trigger object *)
    NEW(platoon, 1..3);
    FOR i := 1 TO 3
        NEW(plt);
        REPLACE(pltID,2,2,INTTOSTR(i));
        ASK plt TO PltInit(SELF,pltID);
        platoon[i] := plt;
    END FOR;
END METHOD; (* CompanyInit *)

(* ----- *)

TELL METHOD ExecuteMovementPlan;
BEGIN
    WHILE ORD(location.symbol) < ORD(ASLTSPN)

```

```

    IF walkingThru
        OUTPUT("Company ",unitName," currently in ",location.symbol);
        MicroDelay(500000);
    END IF;
    WAIT FOR SELF TO MoveTo(location.nextPosition, Foot)
    END WAIT;
END WHILE;
FOR i := 1 TO 3
    ASK platoon[i] TO SetLocation(location);
    WAIT FOR platoon[i] TO
        OccupyFiringPosition(location.firingPositions[i])
    END WAIT;
END FOR;
IF walkingThru
    OUTPUT("Company ",unitName," is in the ",location.symbol);
END IF;
AssignTargets;
END METHOD; (* ExecuteMovementPlan *)

```

(\* ----- \*)

```

TELL METHOD AssignTargets;
VAR
    numTgtsInPltList,
    closestPlt,
    nextClosestPlt,
    farthestPlt,
    j, k, numIn,
    shortestDistance,
    farthestDistance      : INTEGER;
    distToFarthest,
    distOut                : REAL;
    farthestTarget, target : EnemyVehicleObj;
    chosen                  : ARRAY INTEGER OF INTEGER;
    distance                : ARRAY INTEGER OF INTEGER;
    PltTargetList          : ARRAY INTEGER OF StackObj;

BEGIN
    NEW(PltTargetList, 1..3);
    NEW(PltTargetList[1]);
    NEW(PltTargetList[2]);
    NEW(PltTargetList[3]);
    NEW(chosen, 1..3);
    NEW(distance, 1..3);
    numIn := ASK targetList numberIn;
    numTgtsInPltList := CEIL(FLOAT(numIn)/3.0);
    WHILE ASK targetList numberIn > 0
        numIn := ASK targetList numberIn;
        target := ASK targetList First();
        distToFarthest := Distance(location.coordinate, ASK target
                                                                    location);

        farthestTarget := target;

    (* find the target farthest away ... *)

    IF numIn > 1

```

```

FOR k := 1 TO (numIn - 1)
    target := ASK targetList Next(target);
    distOut := Distance(location.coordinate, ASK target location);
    IF distOut > distToFarthest
        farthestTarget := target;
        distToFarthest := distOut;
    END IF;
END FOR;
END IF;

(* ... and assign it to the closest platoon. *)

FOR j := 1 TO 3
    chosen[j] := j;
    distance[j] := ROUND(Distance(ASK platoon[j]
                                location.coordinate, ASK farthestTarget
                                location));
END FOR;
shortestDistance := MINOF(distance[1], distance[2], distance[3]);
farthestDistance := MAXOF(distance[1], distance[2], distance[3]);
IF shortestDistance = distance[1]
    closestPlt := 1;
    chosen[1] := 100;
ELSIF shortestDistance = distance[2]
    closestPlt := 2;
    chosen[2] := 100;
ELSE
    closestPlt := 3;
    chosen[3] := 100;
END IF;
IF farthestDistance = distance[1]
    farthestPlt := 1;
    chosen[1] := 100;
ELSIF farthestDistance = distance[2]
    farthestPlt := 2;
    chosen[2] := 100;
ELSE
    farthestPlt := 3;
    chosen[3] := 100;
END IF;
nextClosestPlt := MINOF(chosen[1], chosen[2], chosen[3]);
IF ASK PltTargetList[nextClosestPlt] numberIn < numTgtsInPltList
    ASK PltTargetList[nextClosestPlt] TO Add(farthestTarget);
    IF walkingThru
        OUTPUT("platoon ", closestPlt, " gets tgt ", ASK farthestTarget
               idNumber);
    END IF;
ELSIF ASK PltTargetList[closestPlt] numberIn < numTgtsInPltList
    ASK PltTargetList[closestPlt] TO Add(farthestTarget);
    IF walkingThru
        OUTPUT("platoon ", closestPlt, " gets tgt ", ASK
               farthestTarget idNumber);
    END IF;
ELSE
    ASK PltTargetList[farthestPlt] TO Add(farthestTarget);
    IF walkingThru

```

```

        OUTPUT("platoon ",farthestPlt," gets tgt ",ASK
                farthestTarget idNumber);
    END IF;
END IF;
ASK targetList TO RemoveThis(farthestTarget);
END WHILE;
IF walkingThru
    Delay(3);
    ClearScreen;
END IF;
FOR i := 1 TO 3
    ASK platoon[i] TO PrepareToEngage(PltTargetList[i]);
END FOR;
DISPOSE(chosen);
DISPOSE(distance);
END METHOD;      (* AssignTargets *)

(* ----- *)

TELL METHOD Hold(IN target : EnemyVehicleObj;
                IN firingPosition : STRING);

BEGIN
    IF NOT alreadyFired
        WAIT FOR movementComplete TO Fire
            alreadyFired := TRUE;
        WAIT DURATION RegroupTime
            TargetHandover(target,firingPosition);
        END WAIT;
    END WAIT;
    ELSE
        WAIT DURATION RegroupTime
            TargetHandover(target,firingPosition);
        END WAIT;
    END IF;
END METHOD;      (* Hold *)

(* ----- *)

ASK METHOD UpdateStatus;
VAR
    readyToMove : BOOLEAN;

BEGIN
    readyToMove := FALSE;
    FOR i := 1 TO 3
        IF NOT ASK platoon[i] set
            set := FALSE;
            EXIT;
        ELSE
            set := TRUE;
        END IF;
    END FOR;

    IF set AND NOT finalAssault

```

```

    ASK myHQ TO UpdateStatus;
END IF;

IF location.symbol = ASLTPSN
    FOR i := 1 TO 3
        IF NOT ASK platoon[i] engagementComplete
            readyToMove := FALSE;
            EXIT;
        ELSE
            readyToMove := TRUE;
        END IF;
    END FOR;
END IF;

IF (readyToMove) AND (NOT finalAssault)
    finalAssault := TRUE;
    FinalAssault;
END IF;
END METHOD;  (* Company UpdateStatus *)

(* ----- *)

TELL METHOD Attack;
BEGIN
    FOR i := 1 TO 3
        TELL platoon[i] TO Engage;
    END FOR;
END METHOD;  (* Attack *)

(* ----- *)

TELL METHOD FinalAssault;
BEGIN
    WAIT FOR SELF TO MoveTo(location.nextPosition, Foot)
    IF walkingThru
        IF firstTimeSet
            Delay(5);
            ClearScreen;
            firstTimeSet := FALSE;
        END IF;
    END IF;
    FOR i := 1 TO 3
        ASK platoon[i] TO SetLocation(location);
        WAIT FOR platoon[i] TO
            OccupyFiringPosition(location.firingPositions[i])
        END WAIT;
    END FOR;
    FOR i := 1 TO 3
        TELL platoon[i] TO FinalAssault;
    END FOR;
END WAIT;
IF walkingThru
    OUTPUT("Company ",unitName," now in ",location.symbol);
END IF;
TELL movementComplete TO Trigger;
END METHOD;  (* FinalAssault *)

```

(\* ----- \*)

```
TELL METHOD TargetHandover(IN target : EnemyVehicleObj;
                          IN firingPosition : STRING);
VAR
  j, closestPlt      : INTEGER;
  dist, shortestDist : REAL;
  handedOver,
  candidate,
  ammoAvail,
  pltInRange          : BOOLEAN;
  pltTargetList       : StackObj;
  unassignableTarget : EnemyVehicleObj;
  Ammo, Busy          : ARRAY INTEGER OF BOOLEAN;

BEGIN
  NEW(pltTargetList);
  NEW(Ammo, 1..3);
  NEW(Busy, 1..3);
  handedOver := FALSE;
  pltInRange := FALSE;
  candidate  := FALSE;
  ammoAvail  := FALSE;
  ASK pltTargetList TO Add(target);
  FOR i := 1 TO 3
    ASK platoon[i] TO UpdateStatus;
  END FOR;
  WAIT DURATION RegroupTime; (* until all platoons complete firing *)
  FOR i := 1 TO 3
    IF NOT ASK platoon[i] outOfATGMammo
      Ammo[i] := TRUE;
      ammoAvail := TRUE;
    ELSE
      Ammo[i] := FALSE;
    END IF;
    IF (NOT ASK platoon[i] engaging) AND
      (NOT ASK platoon[i] outOfATGMammo)
      Busy[i] := FALSE;
      candidate := TRUE;
    ELSE
      Busy[i] := TRUE;
    END IF;
  END FOR;
  IF ammoAvail
    IF candidate
      FOR i := 1 TO 3
        IF (NOT Busy[i]) AND (Ammo[i])
          dist := Distance(ASK platoon[i] location.coordinate,
                          ASK target location);
          IF dist < 1000.0
            IF walkingThru
              OUTPUT("Company ",unitName," handing over target ",ASK
                    target idNumber);
              OUTPUT(" to platoon ",i);
              ASK EngagementHistory TO WriteString("Handing over ");
            
```



```

        ASK EngagementHistory TO WriteInt(ASK target
                                         idNumber,3);
        ASK EngagementHistory TO WriteString(" to platoon ");
        ASK EngagementHistory TO WriteString(ASK platoon[i]
                                         identity);

        ASK EngagementHistory TO WriteLn;
    END IF;
    ASK platoon[i] TO PrepareToEngage(pltTargetList);
    handedOver := TRUE;
    pltInRange := TRUE;
    EXIT;
END IF;
END IF;
END FOR;
IF NOT pltInRange
(* Since no platoon is currently in range, find the closest platoon
and move it to the firing position. *)
shortestDist := 5000.0; (*arbitrary starting distance*)
FOR i := 1 TO 3
    IF (NOT Busy[i]) AND (Ammo[i])
        dist := Distance(ASK platoon[i] location.coordinate,
                        firingPosition);
        IF dist < shortestDist
            closestPlt := i;
            shortestDist := dist;
        END IF;
    END IF;
END FOR;
IF NOT (closestPlt = 0)
WAIT FOR platoon[closestPlt] TO OccupyFiringPosition(firingPosition);
    ASK platoon[closestPlt] TO PrepareToEngage(pltTargetList);
    IF walkingThru
        OUTPUT("Moving platoon ",ASK platoon[closestPlt] identity);
        OUTPUT(" to new position to engage ",ASK target idNumber);
    END IF;
    handedOver := TRUE;
END WAIT;
ELSE
    Hold(target, firingPosition);
END IF;
END IF;
ELSE
    Hold(target, firingPosition);
END IF;
ELSE
    IF walkingThru
        OUTPUT("Unable to handover target ",target.idNumber);
        ASK EngagementHistory TO WriteString("Unable to handover ");
        ASK EngagementHistory TO WriteString("target ");
        ASK EngagementHistory TO WriteInt(target.idNumber,4);
        ASK EngagementHistory TO WriteLn;
    END IF;
    unassignableTarget := ASK pltTargetList TO Remove();
END IF;
END WAIT;
DISPOSE(Ammo); DISPOSE(Busy);

```

```

END METHOD; (* TargetHandover *)

(* ----- *)

TELL METHOD ArtilleryInterrupt(IN casualtyAssessment : REAL);
BEGIN
  IF walkingThru
    OUTPUT("Company ",unitName," receiving fires vicinity ",
           location.symbol);
    OUTPUT(" ...Casualty assessment is ",casualtyAssessment);
    Delay(2);
  END IF;
  IF moving
    Interrupt(SELF, "MoveTo");
    FOR i := 1 TO 3
      ASK platoon[i] TO TakeCasualties(casualtyAssessment);
    END FOR;
  ELSE
    FOR i := 1 TO 3
      TELL platoon[i] TO InterruptEngage;
      ASK platoon[i] TO TakeCasualties(casualtyAssessment);
    END FOR;
  END IF;
END METHOD; (* ArtilleryInterrupt *)

END OBJECT; (* RifleCompanyObj *)

(* ***** *)
OBJECT BattalionObj;
(* ***** *)

ASK METHOD ObjInit;
VAR
  co : RifleCompanyObj;

BEGIN
  NEW(execute); (* Trigger Object *)
  NEW(com,any, A..D);
  FOR name := A TO C
    NEW(co);
    ASK co TO CompanyInit(SELF, name);
    company[name] := co;
  END FOR;
END METHOD; (* ObjInit *)

(* ----- *)

TELL METHOD ExecuteMission;
BEGIN
  FOR name := A TO C
    TELL company[name] TO ExecuteMovementPlan;
  END FOR;
  IF playingArty
    TELL company[A] TO ArtilleryInterrupt(Pk[A]) IN ImpactTimeA;
    TELL company[B] TO ArtilleryInterrupt(Pk[B]) IN ImpactTimeB;
    TELL company[C] TO ArtilleryInterrupt(Pk[C]) IN ImpactTimeC;
  
```

```

END IF;
WAIT FOR execute TO Fire      (* Update status releases *)
    (* To execute a simultaneous attack.... *)
WeaponsStatus := FREE;
firstTimeSet := TRUE;
IF walkingThru
    OUTPUT("Executing a synchronized attack.");
    Delay(2);
    ClearScreen;
END IF;
FOR name := A TO C
    TELL company[name] TO Attack;
END FOR;
END WAIT;
END METHOD;  (* ExecuteMission *)

(* ----- *)

ASK METHOD UpdateStatus;
VAR
    allUnitsSet : BOOLEAN;

BEGIN
    allUnitsSet := FALSE;
    FOR name := A TO C
        IF NOT ASK company[name] set
            allUnitsSet := FALSE;
            EXIT;
        ELSE
            allUnitsSet := TRUE;
        END IF;
    END FOR;

    IF allUnitsSet
        TELL execute TO Trigger;
    END IF;

END METHOD;  (* Battalion UpdateStatus *)

END OBJECT;

END MODULE.

```

## D. FPC

DEFINITION MODULE FPC;

FROM GrpMod     IMPORT StackObj;  
FROM Unit       IMPORT UnitObj;  
FROM OPFOR      IMPORT EnemyVehicleObj;

TYPE

TrooperType = (rifleman, autorifleman, grenadier, machinegunner,  
                  dragongunner, leader);

StrengthList = ARRAY TrooperType OF INTEGER;

ATGMList = ARRAY INTEGER OF ANYOBJ;  
          (\* ARRAY INTEGER OF ATGMObj \*)

FPCObj = OBJECT;       (\* Generic rifle platoon firepower capability \*)  
  myHQ                 : UnitObj;  
  identity,  
  location             : STRING;  
  engaging,  
  ready,  
  outOfATGMammo,  
  firingComplete,  
  finalAssault         : BOOLEAN;  
  strength             : StrengthList;  
  missileSection       : ATGMList;  
  ASK METHOD FPCInit(IN HQ : UnitObj);  
  ASK METHOD DecrementFPC(IN lossPercentage : REAL);  
  ASK METHOD UpdateStatus;  
  ASK METHOD SetLocation(IN coordinate : STRING);  
  TELL METHOD PrepareToFire(IN pltTargetList : StackObj);  
  TELL METHOD ReAssign(IN target : EnemyVehicleObj);  
  TELL METHOD Fire;  
  TELL METHOD FinalAssault;  
  TELL METHOD InterruptFire;  
END OBJECT;

END MODULE.

# IMPLEMENTATION MODULE FPC;

```
FROM GrpMod  IMPORT StackObj;
FROM Unit    IMPORT UnitObj;
FROM ATGM    IMPORT ATGMOBJ;
FROM OPFOR   IMPORT EnemyVehicleObj;
FROM Menu    IMPORT walkingThru;
FROM Globals IMPORT RandomCasualty, ALL TargetStatusType,
EngagementHistory, RegroupTime, AttritionFile;
```

OBJECT FPCObj;

ASK METHOD FPCInit(IN HQ : UnitObj);

VAR

```
  i          : INTEGER;
  dragon     : ATGMOBJ;
  tempId     : STRING;
```

BEGIN

```
  myHQ              := HQ;
  identity          := ASK myHQ identity;
  location          := ASK myHQ location.coordinate;
  engaging          := FALSE;
  ready             := FALSE;
  outOfATGMammo     := FALSE;
  firingComplete    := FALSE;
  finalAssault      := FALSE;
  NEW(strength, rifleman..leader);
  strength[rifleman] := 11;
  strength[autorifleman] := 6;
  strength[grenadier] := 6;
  strength[machinegunner] := 2;
  strength[dragongunner] := 2;
  strength[leader] := 12;
  tempId            := identity;
  NEW(missileSection, 1..strength[dragongunner]);
  FOR i := 1 TO strength[dragongunner]
  NEW(dragon);
    REPLACE(tempId,3,3,INTTOSTR(i));
    ASK dragon TO ATGMInit(SELF, tempId);
    missileSection[i] := dragon;
  END FOR;
END METHOD;          (* FPCInit *)
```

(\* ----- \*)

ASK METHOD DecrementFPC(IN lossPercentage : REAL);

VAR

```
  j          : TrooperType;
  i, numSoldiers,
  numLosses,
  dragonLosses : INTEGER;
  loss, runningSum : REAL;
  dragon       : ATGMOBJ;
```

```

BEGIN
  dragonLosses := 0;
  numSoldiers := 1;
  FOR j := rifleman TO leader
    numSoldiers := numSoldiers + strength[j];
  END FOR;
  numLosses := TRUNC(FLOAT(numSoldiers - 1) * lossPercentage);
  FOR i := 1 TO numLosses
    numSoldiers := numSoldiers - 1;
    j := rifleman;
    runningSum := FLOAT(strength[j])/FLOAT(numSoldiers);
    loss := ASK RandomCasualty Sample();
    LOOP
      IF loss < runningSum
        strength[j] := strength[j] - 1;
        IF j = dragongunner
          INC(dragonLosses);
        END IF;
        EXIT;
      ELSE
        INC(j);
        runningSum := runningSum + FLOAT(strength[j])/FLOAT(numSoldiers);
      END IF;
    END LOOP;
  END FOR;
  IF dragonLosses > 0
    IF dragonLosses = 2
      outOfATGMammo := TRUE;
      ready := TRUE;
      engaging := FALSE;
      firingComplete := TRUE;
      ASK myHQ TO UpdateStatus;
    END IF;
    IF walkingThru
      OUTPUT(" ",identity," dragon losses = ",dragonLosses);
    END IF;
    FOR i := 2 DOWNT0 (3 - dragonLosses)
      dragon := missileSection[i];
      IF NOT (ASK dragon assignedTarget = NILOBJ)
        ReAssign(ASK dragon assignedTarget);
      END IF;
    END FOR;
  END IF;
  IF walkingThru
    ASK AttritionFile TO WriteString("Attrition to platoon " + identity);
    ASK AttritionFile TO WriteString(" with ");
    ASK AttritionFile TO WriteInt(numLosses,4);
    ASK AttritionFile TO WriteString(" losses.");
    ASK AttritionFile TO WriteLn;
    ASK AttritionFile TO WriteString("Strengths for each class of soldier");
    ASK AttritionFile TO WriteLn;
    FOR j := rifleman TO leader
      CASE j
        WHEN rifleman :
          ASK AttritionFile TO WriteString("rifleman ");

```

```

        WHEN autorifleman :
            ASK AttritionFile TO WriteString("autorifleman ");
        WHEN grenadier :
            ASK AttritionFile TO WriteString("grenadier ");
        WHEN machinegunner :
            ASK AttritionFile TO WriteString("machinegunner ");
        WHEN leader :
            ASK AttritionFile TO WriteString("leader ");
        OTHERWISE
            ASK AttritionFile TO WriteString("dragongunner ");
        END CASE;
        ASK AttritionFile TO WriteInt(strength[j],3);
        ASK AttritionFile TO WriteLn;
    END FOR;
    END IF;
END METHOD;          (* DecrementFPC *)

```

(\* ----- \*)

```

ASK METHOD SetLocation(IN coordinate : STRING);
VAR
    i      : INTEGER;
    dragon : ATGMOBJ;

```

```

BEGIN
    location := coordinate;
    IF strength[dragongunner] > 0
        FOR i := 1 TO strength[dragongunner]
            dragon := missileSection[i];
            ASK dragon TO SetLocation(coordinate);
        END FOR;
    END IF;
END METHOD;          (* SetLocation *)

```

(\* ----- \*)

```

ASK METHOD UpdateStatus;
VAR
    i      : INTEGER;
    dragon : ATGMOBJ;

```

```

BEGIN
    IF strength[dragongunner] > 0
        IF NOT finalAssault
            FOR i := 1 TO strength[dragongunner]
                dragon := missileSection[i];
                IF NOT ASK dragon ready
                    ready := FALSE;
                    EXIT;
                ELSE
                    ready := TRUE;
                END IF;
            END FOR;
        END IF;
    END IF;
END METHOD;

```

```

        FOR i := 1 TO strength[dragongunner]
            dragon := missileSection[i];
            IF NOT ASK dragon firingComplete
                firingComplete := FALSE;
                EXIT;
            ELSE
                firingComplete := TRUE;
            END IF;
        END FOR;

        IF ready OR firingComplete
            ASK myHQ TO UpdateStatus;
        END IF;

    END IF;

    FOR i := 1 TO strength[dragongunner]
        dragon := missileSection[i];
        IF ASK dragon missile.ammoCount > 0
            outOfATGMammo := FALSE;
            EXIT;
        ELSE
            outOfATGMammo := TRUE;
        END IF;
    END FOR;

    FOR i := 1 TO strength[dragongunner]
        dragon := missileSection[i];
        IF ASK dragon engaging
            engaging := TRUE;
            EXIT;
        ELSE
            engaging := FALSE;
        END IF;
    END FOR;

    IF (outOfATGMammo) OR (NOT engaging)
        ASK myHQ TO UpdateStatus;
    END IF;

    END IF;
END METHOD;      (* UpdateStatus *)

(* ----- *)

TELL METHOD PrepareToFire(IN pltTargetList : StackObj);
VAR
    i, j, numTargets : INTEGER;
    dragon            : ATGMObj;
    tgt               : EnemyVehicleObj;
    passed            : BOOLEAN;

BEGIN
    j := 1;
    numTargets := ASK pltTargetList numberIn;
    FOR i := 1 TO numTargets

```



```

passed := FALSE;
tgt := ASK pltTargetList TO Remove();
IF j > strength[dragongunner]
    TELL myHQ TO TargetHandover(tgt, location);
    passed := TRUE;
ELSE
    LOOP
        dragon := missileSection[j];
        IF ASK dragon missile.ammoCount > 0
            TELL dragon TO Target(tgt);
            passed := TRUE;
            engaging := TRUE;
            IF (numTargets = 1) AND (j < strength[dragongunner])
                dragon := missileSection[j+1];
                TELL dragon TO Wait;
            END IF;
            EXIT;
        END IF;
        INC(j);
        IF j > strength[dragongunner]
            EXIT;
        END IF;
    END LOOP;
    IF NOT passed
        TELL myHQ TO TargetHandover(tgt, location);
    END IF;
    INC(j);
END FOR;
END METHOD; (* PrepareToFire *)

```

(\* ----- \*)

```

TELL METHOD ReAssign(IN target : EnemyVehicleObj);
VAR
    i          : INTEGER;
    reassigned : BOOLEAN;
    dragon      : ATGMObj;

BEGIN
    reassigned := FALSE;
    WAIT DURATION RegroupTime
    UpdateStatus;
    IF NOT outOfATGMammo
        FOR i := 1 TO strength[dragongunner]
            dragon := missileSection[i];
            IF (ASK dragon missile.ammoCount > 0) AND
                (ASK dragon targetStatus = killed)
                IF walkingThru
                    OUTPUT("Reassigning ",target.idNumber," to ",dragon.identity);
                    ASK EngagementHistory TO WriteString("Reassigning ");
                    ASK EngagementHistory TO WriteInt(ASK target idNumber,3);
                    ASK EngagementHistory TO WriteString(" to ");
                    ASK EngagementHistory TO WriteString(ASK dragon
                        identity);
                    ASK EngagementHistory TO WriteLn;
                END IF;
            END IF;
        END FOR;
    END IF;
END METHOD;

```

```

        END IF;
        TELL dragon TO Target(target);
        engaging := TRUE;
        reassigned := TRUE;
        EXIT;
    END IF;
END FOR;
IF NOT reassigned
    IF walkingThru
        OUTPUT(identity," handing over ",ASK target idNumber);
    END IF;
    TELL myHQ TO TargetHandover(target, location);
END IF;
ELSE
    IF walkingThru
        OUTPUT(identity," handing over ",ASK target idNumber);
    END IF;
    TELL myHQ TO TargetHandover(target, location);
END IF;
END WAIT;
END METHOD;          (* ReAssign *)

```

(\* ----- \*)

```

TELL METHOD Fire;
VAR
    i      : INTEGER;
    dragon : ATGMObj;

BEGIN
    IF strength[dragongunner] > 0
        FOR i := 1 TO strength[dragongunner]
            dragon := missileSection[i];
            TELL dragon TO Fire;
        END FOR;
    END IF;
END METHOD;          (* Fire *)

```

(\* ----- \*)

```

TELL METHOD FinalAssault;
VAR
    i      : INTEGER;
    dragon : ATGMObj;

BEGIN
    finalAssault := TRUE;
    IF strength[dragongunner] > 0
        FOR i := 1 TO strength[dragongunner]
            dragon := missileSection[i];
            IF (ASK dragon targetStatus <> killed)
                AND (NOT ASK dragon unassigned)
                TELL dragon TO EngageArmorTarget;
            END IF;
        END FOR;
    END IF;
END METHOD;

```

```

END METHOD;          (* FinalAssault *)

(* ----- *)

TELL METHOD InterruptFire;
VAR
  i      : INTEGER;
  dragon : ATGMOBJ;

BEGIN
  IF strength[dragongunner] > 0
    FOR i := 1 TO strength[dragongunner]
      dragon := missileSection[i];
      TELL dragon TO InterruptMissileFire;
    END FOR;
  END IF;
END METHOD;          (* InterruptFire *)

END OBJECT;          (* FPCObj *)

END MODULE.

```

## E. ATGM

### DEFINITION MODULE ATGM:

```
FROM SimMod      IMPORT TriggerObj;
FROM RandMod     IMPORT RandomObj;
FROM FPC         IMPORT FPCObj;
FROM OPFOR       IMPORT EnemyVehicleObj;
FROM Weapons     IMPORT MissileRecordType;
FROM Globals     IMPORT TargetStatusType;
```

### TYPE

```
ATGMObj = OBJECT
  myUnit          : FPCObj;
  identity        : STRING;
  location        : STRING;
  missile         : MissileRecordType;
  permission      : TriggerObj;
  assignedTarget  : EnemyVehicleObj;
  distanceToTarget : REAL;
  targetStatus    : TargetStatusType;
  engaging,
  unassigned,
  acquired,
  ready,
  tracking,
  firingComplete  : BOOLEAN;
  ASK METHOD ATGMInit(IN unit : FPCObj;
                     IN id   : STRING);
  ASK METHOD UpdateMissileStatus;
  ASK METHOD SetLocation(IN coordinate : STRING);
  TELL METHOD Target(IN target : EnemyVehicleObj);
  TELL METHOD Wait;
  TELL METHOD EngageArmorTarget;
  TELL METHOD PrepMissile;
  TELL METHOD AcquireTarget;
  TELL METHOD Fire;
  TELL METHOD TrackMissile;
  TELL METHOD CutWires;
  TELL METHOD InterruptMissileFire;
END OBJECT;
```

END MODULE.

# IMPLEMENTATION MODULE ATGM;

```

FROM SimMod      IMPORT Interrupt;
FROM OPFOR       IMPORT EnemyVehicleObj;
FROM FPC         IMPORT FPCObj;
FROM MapRecon    IMPORT Distance;
FROM Impact     IMPORT ALL ImpactAreaType, AspectAngle, AssessDamage;
FROM Weapons     IMPORT ALL MissileType, MissileSystem;
FROM Menu        IMPORT walkingThru;
FROM MOE         IMPORT TotalOPFORlosses;
FROM Globals     IMPORT ALL WeaponsStatusType, OutputFile,
                        EngagementHistory, WeaponsStatus,
                        ALL TargetStatusType, HitOrMiss;

```

OBJECT ATGMObj;

```

ASK METHOD ATGMInit(IN unit : FPCObj;
                   IN id   : STRING);

```

BEGIN

```

    myUnit      := unit;
    identity    := id;
    location    := ASK myUnit location;
    missile     := CLONE(MissileSystem[Dragon]);
    assignedTarget := NILOBJ;
    unassigned  := FALSE;
    targetStatus := missed;
    acquired    := FALSE;
    ready       := FALSE;
    tracking     := FALSE;
    firingComplete := FALSE;
    engaging    := FALSE;
    NEW(permission);
END METHOD; (* ATGMInit *)

```

(\* ----- \*)

```

ASK METHOD UpdateMissileStatus;

```

(\* PrepMissile and AcquireTarget invoke this method when  
their status changes \*)

BEGIN

```

    IF acquired
        ready := TRUE;
        IF WeaponsStatus = HOLD
            ASK myUnit TO UpdateStatus;
        ELSE
            TELL permission TO Trigger;
        END IF;
    END IF;
END METHOD; (* UpdateMissileStatus *)

```

(\* ----- \*)

```

ASK METHOD SetLocation(IN coordinate : STRING);
BEGIN

```

```

    location := coordinate;
END METHOD;  (* SetLocation *)

```

```

(* ----- *)

```

```

TELL METHOD Target(IN target : EnemyVehicleObj);
BEGIN
    unassigned      := FALSE;
    assignedTarget := target;
    targetStatus    := missed;
    engaging        := TRUE;
    IF missile.ammoCount > 0
        EngageArmorTarget;
    ELSE
        TELL myUnit TO ReAssign(target);
    END IF;
END METHOD;  (* Target *)

```

```

(* ----- *)

```

```

TELL METHOD Wait;
BEGIN
    engaging      := FALSE;
    unassigned    := TRUE;
    ready         := TRUE;
    targetStatus  := killed;
    firingComplete := TRUE;
    assignedTarget := NILOBJ;
    ASK myUnit TO UpdateStatus;
END METHOD;  (* Wait *)

```

```

(* ----- *)

```

```

TELL METHOD EngageArmorTarget;

```

```

(* This method simulates an ATGM (Dragon/TOW) engagement. The
   gunner receives a fire mission, prepares the missile, acquires
   the target, fires, and tracks the missile until impact or
   interrupted by incoming fires. *)

```

```

BEGIN

```

```

(* The WAIT FOR is used below so that any methods waiting will also
   terminate if one is interrupted. *)

```

```

    WAIT FOR SELF TO PrepMissile;
    END WAIT;

```

```

    WAIT FOR permission TO Fire (*from UpdateMissileStatus or Fire*)

```

```

    IF distanceToTarget <= missile.maxEffRange

```

```

        WAIT FOR SELF TO TrackMissile
        END WAIT;

```

```

    ELSIF ASK myUnit finalAssault

```

```

        Wait; (* moved out of range of previously assigned target *)

```

```

        IF walkingThru

```

```

            ASK EngagementHistory TO WriteString(identity);

```

```

            ASK EngagementHistory TO WriteString(" moved out of ");

```

```

            ASK EngagementHistory TO WriteString(" range of target ");

```

```

            ASK EngagementHistory TO

```

```

                WriteInt(assignedTarget.idNumber,3);

```

```

        ASK EngagementHistory TO WriteLn;
    END IF;
ELSE
    acquired      := FALSE;
    ready         := FALSE;
    firingComplete := TRUE;
END IF;
    ASK myUnit TO UpdateStatus;
ON INTERRUPT
    TERMINATE;
END WAIT;
END METHOD;  (* EngageArmorTarget *)

```

(\* ----- \*)

```

TELL METHOD PrepMissile;
BEGIN
    WAIT DURATION missile.prepTime
    AcquireTarget;
    ON INTERRUPT (* Take cover! Incoming fires... *)
        TERMINATE;
    END WAIT;
END METHOD;  (* PrepMissile *)

```

(\* ----- \*)

```

TELL METHOD AcquireTarget;
BEGIN
    WAIT DURATION missile.acquisitionTime
    IF assignedTarget <> NILOBJ
        distanceToTarget := Distance(location, ASK assignedTarget
                                     location);
        acquired := TRUE;
        UpdateMissileStatus;
    ELSE
        TERMINATE;
    END IF;
    ON INTERRUPT (* Take cover! Incoming fires... *)
        acquired := FALSE;
        TERMINATE;
    END WAIT;
END METHOD;  (* AcquireTarget *)

```

(\* ----- \*)

```

TELL METHOD Fire;
BEGIN
    TELL permission TO Trigger;
END METHOD;  (* Fire *)

```

(\* ----- \*)

```

TELL METHOD TrackMissile;
VAR
    result      : TargetStatusType;
    region      : ImpactAreaType;

```

```

BEGIN
    tracking := TRUE;
    WAIT DURATION distanceToTarget / missile.velocity (*tracking time*)
    missile.ammoCount := missile.ammoCount - 1;
    tracking := FALSE;
    CutWires;
    IF walkingThru
        ASK EngagementHistory TO WriteString(identity);
    END IF;
    (* sample probability of hit *)
    IF ASK HitOrMiss UniformReal(0.0,1.0) < missile.pHit
        region := AspectAngle(location, assignedTarget);
        result := AssessDamage(missile, assignedTarget, region);
        targetStatus := result;
        IF walkingThru
            OUTPUT(identity, " ", result, " ", ASK assignedTarget
                idNumber);
        END IF;
        CASE result
            WHEN killed :
                TotalOPFORlosses := TotalOPFORlosses + 1;
                engaging := FALSE;
                ASK assignedTarget TO
                    VehicleTerminate(missile.system, ORD(region));
                IF walkingThru
                    ASK EngagementHistory TO WriteString(" killed");
                    ASK EngagementHistory TO WriteInt(ASK
                        assignedTarget idNumber, 3);
                END IF;
                assignedTarget := NILOBJ;
            WHEN damaged:
                IF walkingThru
                    ASK EngagementHistory TO WriteString(" damaged ");
                    ASK EngagementHistory TO WriteInt(ASK
                        assignedTarget idNumber, 3);
                END IF;
                IF missile.ammoCount = 0
                    engaging := FALSE;
                    TELL myUnit TO ReAssign(assignedTarget);
                ELSIF ASK myUnit finalAssault
                    EngageArmorTarget;
                END IF;
        END CASE;
    ELSE
        targetStatus := missed;
        IF walkingThru
            OUTPUT(identity, " missed ", ASK assignedTarget idNumber);
            ASK EngagementHistory TO WriteString(" missed ");
            ASK EngagementHistory TO WriteInt(ASK assignedTarget
                idNumber, 3);
        END IF;
        IF missile.ammoCount = 0
            engaging := FALSE;
            TELL myUnit TO ReAssign(assignedTarget);
        ELSIF ASK myUnit finalAssault

```



```

        EngageArmorTarget;
    END IF;
END IF;
IF walkingThru
    ASK EngagementHistory TO WriteLn;
END IF;
ON INTERRUPT (* Take Cover! Incoming fires... *)
    DEC(missile.ammoCount); (* lost missile *)
    IF walkingThru
        OUTPUT(identity," lost missile during artillery strike ");
    END IF;
    tracking := FALSE;
    IF missile.ammoCount = 0
        engaging := FALSE;
        TELL myUnit TO ReAssign(assignedTarget);
    END IF;
    TERMINATE;
END WAIT;
END METHOD; (* TrackMissile *)

(* ----- *)

TELL METHOD CutWires;
BEGIN
    (* elapse time to dismount Dragon sight or cut TOW wires *)
    WAIT DURATION missile.cutTime
        acquired      := FALSE;
        ready         := FALSE;
        firingComplete := TRUE;
        ASK myUnit TO UpdateStatus;
    END WAIT;
END METHOD; (* CutWires *)

(* ----- *)

TELL METHOD InterruptMissileFire;

(* called from higher unit receiving indirect fires *)

BEGIN
    Interrupt(SELF,"PrepMissile");
    Interrupt(SELF,"AcquireTarget");
    Interrupt(SELF,"TrackMissile");
    ASK myUnit TO UpdateStatus;
END METHOD; (* InterruptMissileFire *)

END OBJECT; (* ATGMOject *)

END MODULE.

```

## F. MAPRECON

DEFINITION MODULE MapRecon;

FROM GrpMod IMPORT StackObj;  
FROM Globals IMPORT UnitNameType;

TYPE

SymbolType = (ATKPSN, LD, CP1, CP2, CP3, CP4, CP5, CP6,  
CP7, CP8, CP9, CP10, ASLTPSN, INTOBJ, OBJ);

TargetList = ARRAY UnitNameType OF StackObj;

PositionRecordType = RECORD  
symbol : SymbolType;  
coordinate : STRING;  
firingPositions : ARRAY INTEGER OF STRING;  
nextPosition : PositionRecordType;  
END RECORD;

UnitMovementRouteList = ARRAY UnitNameType OF PositionRecordType;

PROCEDURE ModelOperationsOverlay;

PROCEDURE Distance(IN coord1, coord2 : STRING): REAL;

VAR

position : PositionRecordType;  
UnitRoute : UnitMovementRouteList;  
UnitTargetList : TargetList;

END MODULE.

IMPLEMENTATION MODULE MapRecon;

```
FROM MathMod IMPORT Sqrt, Ceil;
FROM GrpMod  IMPORT StackObj;
FROM IOMod   IMPORT StreamObj, FileUseType(Input);
FROM OPFOR   IMPORT EnemyVehicleRef;
FROM Menu    IMPORT selectedModel, walkingThru;
FROM Globals IMPORT ALL UnitNameType;
```

PROCEDURE ModelOperationsOverlay;

VAR

```
  i, numTgts, targetID,
  numFiringPositions,
  symbolCrossReferenceNumber : INTEGER;
  nilentry                  : STRING;
  j                          : UnitNameType;
  coordinate                 : STRING;
  TerrainDataFile           : StreamObj;
  targetList                : StackObj;
  futurePosition            : PositionRecordType;
```

BEGIN

```
  j := A;
  NEW(TerrainDataFile);
  CASE selectedModel
    WHEN 1 : ASK TerrainDataFile TO Open("terrain.dat", Input);
    WHEN 2 : ASK TerrainDataFile TO Open("terrain2.dat", Input);
    OTHERWISE
      ASK TerrainDataFile TO Open("terrain3.dat", Input);
  END CASE;
  ASK TerrainDataFile TO ReadLine(nilentry);
  NEW(UnitRoute, A..D);
  NEW(UnitTargetList, A..D);
  WHILE NOT ASK TerrainDataFile eof
    WHILE j <= C
      ASK TerrainDataFile TO ReadLine(nilentry);
      LOOP
        NEW(position);
        ASK TerrainDataFile TO ReadInt(symbolCrossReferenceNumber);
        position.symbol := VAL(SymbolType, symbolCrossReferenceNumber);
        ASK TerrainDataFile TO ReadString(position.coordinate);
        ASK TerrainDataFile TO ReadInt(numFiringPositions);
        IF numFiringPositions > 0
          NEW(position.firingPositions, 1..numFiringPositions);
          FOR i := 1 TO numFiringPositions
            ASK TerrainDataFile TO
              ReadString(position.firingPositions[i]);
          END FOR;
        END IF;
        ASK TerrainDataFile TO ReadLine(nilentry);
        IF symbolCrossReferenceNumber < ORD(OBJ)
          position.nextPosition := futurePosition;
        END IF;
        futurePosition := position;
        IF symbolCrossReferenceNumber = 0
          EXIT;
        END IF;
      END LOOP;
    END WHILE;
  END WHILE;
```

```

    END IF;
  END LOOP;
  UnitRoute[j] := position;
  ASK TerrainDataFile TO ReadLine(nilentry);
  ASK TerrainDataFile TO ReadLine(nilentry);
  ASK TerrainDataFile TO ReadInt(numTgts);
  IF numTgts > 0
    NEW(UnitTargetList[j]);
    FOR i := 1 TO numTgts
      ASK TerrainDataFile TO ReadInt(targetID);
      ASK UnitTargetList[j] TO
        Add(EnemyVehicleRef[targetID]);
    END FOR;
    ASK TerrainDataFile TO ReadLine(nilentry);
  END IF;
  INC(j);
END WHILE;
END WHILE;
ASK TerrainDataFile TO Close;
DISPOSE(TerrainDataFile);
IF walkingThru
  OUTPUT("Model Operations Overlay complete. ");
  OUTPUT;
END IF;
END PROCEDURE;  (* ModelOperationsOverlay *)

(* ----- *)

PROCEDURE Distance(IN coord1, coord2 : STRING) : REAL;

  (* Given two locations in UTM Grid Coordinates,( note these
  are 6-digit (100meter) coordinates with two letter identifier)
  this subroutine determines the straight-line distance in meters
  between the two points. A critical assumption of this procedure is
  that the two points will, at most, lie on two adjacent map sheets.*)

  VAR
    gridIdentifier1, gridIdentifier2 : STRING;
    Xcoord1, Xcoord2,
      Ycoord1, Ycoord2                : REAL;
    DeltaX, DeltaY                    : REAL;
    northcoord, southcoord,
      eastcoord, westcoord            : REAL;

  BEGIN

    gridIdentifier1 := SUBSTR(1,2,coord1);
    gridIdentifier2 := SUBSTR(1,2,coord2);
    Xcoord1        := STRTOREAL(SUBSTR(3,5,coord1));
    Xcoord2        := STRTOREAL(SUBSTR(3,5,coord2));
    Ycoord1        := STRTOREAL(SUBSTR(6,8,coord1));
    Ycoord2        := STRTOREAL(SUBSTR(6,8,coord2));

    (* The following variables are used when the two points lie on adjacent
    map sheets.  *)

```

```

northcoord      := 1000.0 + MINOF(Ycoord1,Ycoord2);
southcoord      := MAXOF(Ycoord1,Ycoord2);
eastcoord       := 1000.0 + MINOF(Xcoord1,Xcoord2);
westcoord       := MAXOF(Xcoord1,Xcoord2);

IF gridIdentifier1 = gridIdentifier2
  (* Locations are within the same 100,000 square meter grid
    identification zone. *)
  DeltaX := ABS(Xcoord1 - Xcoord2);
  DeltaY := ABS(Ycoord1 - Ycoord2);

ELSIF SCHAR(gridIdentifier1,1) = SCHAR(gridIdentifier2,1)
  (* Locations are in adjacent North-South grid
    identification zones. *)
  DeltaX := ABS(Xcoord1 - Xcoord2);
  DeltaY := northcoord - southcoord;

ELSIF SCHAR(gridIdentifier1,2) = SCHAR(gridIdentifier2,2)
  (* Locations are in adjacent East-West grid
    identification zones. *)
  DeltaX := eastcoord - westcoord;
  DeltaY := ABS(Ycoord1 - Ycoord2);

ELSE
  (* Locations are in diagonally adjacent grid
    identification zones. *)
  DeltaX := eastcoord - westcoord;
  DeltaY := northcoord - southcoord;

END IF;

RETURN (SQRT(DeltaX*DeltaX + DeltaY*DeltaY)) * 100.0;

END PROCEDURE; (* Distance *)

END MODULE.
```

## G. OPFOR

DEFINITION MODULE OPFOR;

FROM RandMod IMPORT RandomObj;  
FROM Weapons IMPORT MissileType;

TYPE

EnemyVehicleType = (BMP, BRDM, T72, ZSU234);

EnemyVehicleObj = OBJECT

idNumber,  
engagementCount : INTEGER;  
type : EnemyVehicleType;  
location : STRING; (\* UTM Grid coordinate \*)  
orientation : INTEGER;  
ASK METHOD ObjInit;  
ASK METHOD VehicleTerminate(IN whatShotMe : MissileType;  
IN where : INTEGER);

END OBJECT;

EnemyVehicleRefList = ARRAY INTEGER OF EnemyVehicleObj;

PROCEDURE ModelEnemyDefense;

VAR

defender : EnemyVehicleObj;  
IDnumber : INTEGER;  
Type : EnemyVehicleType;  
Location : STRING;  
Orientation : INTEGER;  
EnemyVehicleRef : EnemyVehicleRefList;

END MODULE.

# IMPLEMENTATION MODULE OPFOR;

```
FROM SimMod  IMPORT SimTime;
FROM IOMod   IMPORT StreamObj, FileUseType(Input);
FROM Weapons IMPORT ALL MissileType;
FROM Globals IMPORT OutputFile;
FROM MOE     IMPORT TotalOPFORstarting, TotalOPFORlosses;
FROM Menu    IMPORT walkingThru;
```

## PROCEDURE ModelEnemyDefense;

VAR

```
OPFORdataFile      : StreamObj;
enemyVehicleCrossReferenceNumber : INTEGER;
nilentry           : STRING;
```

BEGIN

```
TotalOPFORlosses := 0;
TotalOPFORstarting := 0;
NEW(OPFORdataFile);
ASK OPFORdataFile TO Open("opfor.dat", Input);
ASK OPFORdataFile TO ReadLine(nilentry);
NEW(EnemyVehicleRef, 93..210);
WHILE NOT ASK OPFORdataFile eof
    ASK OPFORdataFile TO ReadInt(IDnumber);
    ASK OPFORdataFile TO ReadInt(enemyVehicleCrossReferenceNumber);
    ASK OPFORdataFile TO ReadString(Location);
    ASK OPFORdataFile TO ReadInt(Orientation);
    ASK OPFORdataFile TO ReadLine(nilentry);
    Type := VAL(EnemyVehicleType, enemyVehicleCrossReferenceNumber);
    NEW(defender);
    TotalOPFORstarting := TotalOPFORstarting + 1;
    EnemyVehicleRef[IDnumber] := CLONE(defender);
END WHILE;
ASK OPFORdataFile TO Close;
DISPOSE(OPFORdataFile);
IF walkingThru
    OUTPUT("Model Enemy Defense complete. ");
    OUTPUT;
END IF;
END PROCEDURE; (* ModelEnemyDefense *)
```

(\* ----- \*)

OBJECT EnemyVehicleObj;

ASK METHOD ObjInit;

BEGIN

```
idNumber      := IDnumber;
type          := Type;
location      := Location;
orientation   := Orientation;
engagementCount := 0;
END METHOD;
```

(\* ----- \*)

```

ASK METHOD VehicleTerminate(IN whatShotMe : MissileType;
                           IN where       : INTEGER);
VAR
    weapon,
    region,
    vehicleType : STRING;
BEGIN
    IF walkingThru
    CASE whatShotMe
        WHEN Dragon : weapon := "dragon";
        OTHERWISE
            weapon := "TOW";
    END CASE;
    CASE where
        WHEN 0      : region := "frontal";
        WHEN 1      : region := "flank";
        OTHERWISE
            region := "rear";
    END CASE;
    CASE type
        WHEN BMP      : vehicleType := "BMP";
        WHEN BRDM     : vehicleType := "BRDM";
        WHEN T72      : vehicleType := "T72";
        OTHERWISE
            vehicleType := "ZSU234";
    END CASE;
    engagementCount := engagementCount + 1;
    IF engagementCount > 1
        ASK OutputFile TO WriteString("Multiply engaged target");
        ASK OutputFile TO WriteInt(engagementCount, 5);
        ASK OutputFile TO WriteString(" engagements thus far.");
    END IF;
    ASK OutputFile TO WriteString("Enemy ");
    ASK OutputFile TO WriteString(vehicleType);
    ASK OutputFile TO WriteString(" number");
    ASK OutputFile TO WriteInt(idNumber,5);
    ASK OutputFile TO WriteString(" KIA.");
    ASK OutputFile TO WriteLn;
    ASK OutputFile TO WriteString(" Killed at H + ");
    ASK OutputFile TO WriteReal(SimTime()/3600.0,4,1);
    ASK OutputFile TO WriteString("hrs by weapon type ");
    ASK OutputFile TO WriteString(weapon);
    ASK OutputFile TO WriteString(" from a ");
    ASK OutputFile TO WriteString(region);
    ASK OutputFile TO WriteString(" shot.");
    ASK OutputFile TO WriteLn;
    ASK OutputFile TO WriteLn;
END IF;
END METHOD;

END OBJECT;

END MODULE.

```



## H. IMPACT

DEFINITION MODULE Impact;

FROM OPFOR IMPORT EnemyVehicleObj;  
FROM Weapons IMPORT MissileRecordType;  
FROM Globals IMPORT TargetStatusType;

TYPE

ImpactAreaType = (front, flank, rear);

PROCEDURE AspectAngle(IN GunLocation : STRING;  
                      IN Target       : EnemyVehicleObj) :  
  ImpactAreaType;

PROCEDURE AssessDamage(IN missile      : MissileRecordType;  
                      IN target       : EnemyVehicleObj;  
                      IN impactPoint : ImpactAreaType) :  
  TargetStatusType;

END MODULE.

IMPLEMENTATION MODULE Impact;

```
FROM MathMod  IMPORT ATAN, ACOS, SIN, COS, pi;
FROM OPFOR    IMPORT EnemyVehicleObj;
FROM Weapons  IMPORT ALL MissileType, MissileRecordType;
FROM Globals  IMPORT ALL TargetStatusType, BDA;
```

```
PROCEDURE AspectAngle(IN GunLocation : STRING ;
                      IN Target      : EnemyVehicleObj) :
                      ImpactAreaType;
```

(\* Given a gun location and a target location in 6-digit (100 m) UTM coordinates with two letter identifier, this procedure determines the engagement aspect angle and returns the region of the target in which the round impacts. This model assumes targets are symmetric with respect to their center of mass.

Calculation of aspect angle is based on vector mathematics, where the aspect angle ALPHA is obtained from the dot product of the gun-target vector GAMMA, and the target orientation vector THETA, where the target location is the origin with Grid North as 0 degrees. \*)

VAR

```
ALPHA, GAMMA, THETA          : REAL;
gridIdentifierGun, gridIdentifierTgt : STRING;
gunXcoord, tgtXcoord,
gunYcoord, tgtYcoord          : INTEGER;
DeltaX, DeltaY                : INTEGER;
northcoord, southcoord,
eastcoord, westcoord          : INTEGER;
```

BEGIN

```
gridIdentifierGun := SUBSTR(1,2,GunLocation);
gridIdentifierTgt := SUBSTR(1,2,ASK Target location);
gunXcoord         := STRTOINT(SUBSTR(3,5,GunLocation));
tgtXcoord         := STRTOINT(SUBSTR(3,5,ASK Target location));
gunYcoord         := STRTOINT(SUBSTR(6,8,GunLocation));
tgtYcoord         := STRTOINT(SUBSTR(6,8,ASK Target location));
```

(\* The following variables are used when the two points lie on adjacent map sheets. \*)

```
northcoord := 1000 + MINOF(gunYcoord,tgtYcoord);
southcoord := MAXOF(gunYcoord,tgtYcoord);
eastcoord  := 1000 + MINOF(gunXcoord,tgtXcoord);
westcoord  := MAXOF(gunXcoord,tgtXcoord);
```

(\* Convert target orientation angle to radians. \*)  
THETA := FLOAT(ASK Target orientation) \* pi / 180.0;

```

(* Determine the horizontal and vertical components of the
gun-target vector. *)

IF gridIdentifierGun = gridIdentifierTgt
  (* Locations are within the same 100,000m square
  identification zone. *)
  DeltaX := ABS(gunXcoord - tgtXcoord);
  DeltaY := ABS(gunYcoord - tgtYcoord);
  (* in this case, the components do not need to be normalized *)
  northcoord := MAXOF(gunYcoord, tgtYcoord);
  eastcoord := MAXOF(gunXcoord, tgtXcoord);

ELSIF SCHAR(gridIdentifierGun,1) = SCHAR(gridIdentifierTgt,1)
  (* Locations are in adjacent North-South grid
  identification zones. *)
  DeltaX := ABS(gunXcoord - tgtXcoord);
  DeltaY := northcoord - southcoord;

ELSIF SCHAR(gridIdentifierGun,2) = SCHAR(gridIdentifierTgt,2)
  (* Locations are in adjacent East-West grid
  identification zones. *)
  DeltaX := eastcoord - westcoord;
  DeltaY := ABS(gunYcoord - tgtYcoord);

ELSE
  (* Locations are in diagonally adjacent grid
  identification zones. *)
  DeltaX := eastcoord - westcoord;
  DeltaY := northcoord - southcoord;

END IF;

(* Now determine the angle, GAMMA, between the gun-target line and
Grid North.
First case...target is north of the gun *)

IF (northcoord = 1000 + tgtYcoord) OR (northcoord = tgtYcoord)
  DeltaY := - DeltaY;
END IF;
(* Second case...target is east of the gun *)
IF (eastcoord = 1000 + tgtXcoord) OR (eastcoord = tgtXcoord)
  DeltaX := - DeltaX;
END IF;

IF DeltaY = 0
  IF DeltaX > 0
    GAMMA := pi/2.0;
  ELSE
    GAMMA := -pi/2.0;
  END IF;
ELSE
  GAMMA := ATAN(FLOAT(DeltaX)/FLOAT(DeltaY));
END IF;

IF ((DeltaY < 0) AND (DeltaX > 0)) (*gun is in the 4th qu d*)
  OR ((DeltaY < 0) AND (DeltaX < 0))(*gun is in the 3rd quad*)

```

```

    GAMMA := GAMMA + pi;
END IF;
IF ((DeltaY > 0) AND (DeltaX < 0)) (*gun is in the 2nd quad*)
    GAMMA := GAMMA + 2.0 * pi;
END IF;

(* Now we can get aspect angle ALPHA *)

ALPHA := ACOS(SIN(GAMMA)*SIN(THETA) + COS(GAMMA)*COS(THETA));

(* The aspect angle identifies one of the three regions of
   impact: front, flank, rear *)

IF (ALPHA >= 7.0*pi/4.0) OR (ALPHA <= pi/4.0)
    RETURN front;
ELSIF (ALPHA >= 3.0*pi/4.0) AND (ALPHA <= 5.0*pi/4.0)
    RETURN rear;
ELSE
    RETURN flank;
END IF;

END PROCEDURE;

(* ----- *)

PROCEDURE AssessDamage(IN missile      : MissileRecordType;
                      IN target       : EnemyVehicleObj;
                      IN impactPoint: ImpactAreaType) :
                      TargetStatusType;
BEGIN
    IF ASK BDA UniformReal(0.0,1.0) <
        missile.pKill[ORD(target.type),ORD(impactPoint)]
        RETURN killed;
    ELSE
        RETURN damaged;
    END IF;
END PROCEDURE;

END MODULE.

```

## I. WEAPONS

DEFINITION MODULE Weapons;

TYPE

MissileType = (Dragon, TOW);

KillProbList = ARRAY INTEGER, INTEGER OF REAL;  
(\* ARRAY EnemyVehicleType, ImpactAreaType... \*)

MissileEffectivenessList = ARRAY MissileType OF KillProbList;

MissileRecordType = RECORD  
    system              : MissileType;  
    velocity,  
    maxEffRange,  
    prepTime,  
    acquisitionTime,  
    cutTime             : REAL;  
    ammoCount           : INTEGER;  
    pHit                : REAL;  
    pKill               : KillProbList;  
END RECORD;

MissileSystemList = ARRAY MissileType OF MissileRecordType;

PROCEDURE ReadMissileData;

VAR

    missile             : MissileRecordType;  
    MissileSystem      : MissileSystemList;  
    KillProb           : KillProbList;  
    MissileEffect      : MissileEffectivenessList;

END MODULE.

IMPLEMENTATION MODULE Weapons;

FROM IOMod IMPORT StreamObj, FileUseType(Input);

PROCEDURE ReadKillProbData;

VAR

i : MissileType;  
j, k : INTEGER;  
nilentry : STRING;  
KillDataFile : StreamObj;

BEGIN

NEW(KillDataFile);

ASK KillDataFile TO Open("pkill.dat", Input);

NEW(MissileEffect, Dragon..TOW);

FOR i := Dragon TO TOW

NEW(KillProb, 0..3, 0..2);

ASK KillDataFile TO ReadLine(nilentry);

FOR j := 0 TO 3

ASK KillDataFile TO ReadString(nilentry);

FOR k := 0 TO 2

ASK KillDataFile TO ReadReal(KillProb[j,k]);

END FOR;

END FOR;

ASK KillDataFile TO ReadLine(nilentry);

MissileEffect[i] := CLONE(KillProb);

DISPOSE(KillProb);

END FOR;

ASK KillDataFile TO Close;

DISPOSE(KillDataFile);

END PROCEDURE; (\* ReadKillProbData \*)

(\* ----- \*)

PROCEDURE ReadMissileData;

VAR

i : MissileType;  
nilentry : STRING;  
WeaponsFile : StreamObj;

BEGIN

ReadKillProbData;

NEW(WeaponsFile);

ASK WeaponsFile TO Open("missile.dat", Input);

ASK WeaponsFile TO ReadLine(nilentry);

NEW(MissileSystem, Dragon..TOW);

FOR i := Dragon TO TOW

NEW(missile);

missile.system := i;

ASK WeaponsFile TO ReadLine(nilentry);

ASK WeaponsFile TO ReadReal(missile.velocity);

ASK WeaponsFile TO ReadReal(missile.maxEffRange);

ASK WeaponsFile TO ReadReal(missile.prepTime);

ASK WeaponsFile TO ReadReal(missile.acquisitionTime);

ASK WeaponsFile TO ReadReal(missile.cutTime);

```
    ASK WeaponsFile TO ReadInt(missile.ammoCount);
    ASK WeaponsFile TO ReadReal(missile.pHit);
    ASK WeaponsFile TO ReadLine(nilentry);
    missile.pKill := MissileEffect[i];
    MissileSystem[missile.system] := missile;
END FOR;
ASK WeaponsFile TO Close;
DISPOSE(WeaponsFile);
END PROCEDURE;

END MODULE.
```

## J. ARTY

DEFINITION MODULE Arty;

FROM Globals IMPORT UnitNameType;

TYPE

    PROCEDURE ScheduleOPFORArty;

VAR

    ImpactTimeA,

    ImpactTimeB,

    ImpactTimeC : REAL;

    Pk : ARRAY UnitNameType OF REAL;

END MODULE.



```

IMPLEMENTATION MODULE Arty;

FROM MathMod IMPORT POWER, EXP, SQRT;
FROM RandMod IMPORT RandomObj, FetchSeed;
FROM Globals IMPORT ALL UnitNameType, RoundGenerator;
FROM Menu     IMPORT selectedModel, walkingThru;

PROCEDURE ScheduleOPFORArty;

CONST
  GunsFiring = 6;
  LethalArea = 1963.495 ; (*  $\pi$  - lethal radius = 25m *)
  TargetArea = 28600.0 ; (* NTC IFCAS Box *)

VAR
  RoundsPerGun : INTEGER;
  Z             : REAL;
  Unit          : UnitNameType;

BEGIN
  CASE selectedModel
    WHEN 1 :   ImpactTimeA := 25200.0;
              ImpactTimeB := 28800.0;
              ImpactTimeC := 3600.0;
    WHEN 2 :   ImpactTimeA := 3600.0;
              ImpactTimeB := 12096.0;
              ImpactTimeC := 3600.0;
    OTHERWISE ImpactTimeA := 5000.0;
              ImpactTimeB := 7200.0;
              ImpactTimeC := 3600.0;
  END CASE;
  NEW(Pk, A..C);
  FOR Unit := A TO C
    RoundsPerGun := ASK RoundGenerator UniformInt(1,3);
    Z := FLOAT(GunsFiring * RoundsPerGun) * LethalArea / TargetArea;
    Pk[Unit] := POWER((1.0 - EXP(-SQRT(Z))),2.0);
  END FOR;
END PROCEDURE;

END MODULE.

```

## K. MOE

DEFINITION MODULE MOE;

TYPE

```
PROCEDURE Mean(IN replicationNumber : INTEGER;
               IN oldAvg           : REAL;
               IN currentSample     : REAL) : REAL;
```

```
PROCEDURE MOEmean(IN replicationNumber : INTEGER;
                  IN currentSample     : REAL);
```

```
PROCEDURE MOEvariance(IN replicationNumber : INTEGER;
                      IN oldAvg           : REAL;
                      IN currentSample     : REAL);
```

PROCEDURE ReportStats;

VAR

```
TotalOPFORlosses,
TotalOPFORstarting      : INTEGER;
MeanMOE, VarianceMOE    : REAL;
meanMissionTime,
meanAttritionForThisRun,
percentAttrition        : REAL;
```

END MODULE.

```

IMPLEMENTATION MODULE MOE;

FROM MathMod  IMPORT POWER;
FROM Globals  IMPORT OutputFile;
FROM Menu     IMPORT numberOfReplications;

PROCEDURE Mean(IN replicationNumber : INTEGER;
               IN oldAvg            : REAL;
               IN currentSample     : REAL) : REAL;

BEGIN
    RETURN ((FLOAT(replicationNumber) * oldAvg) +
            currentSample) / FLOAT(replicationNumber + 1);
END PROCEDURE;

PROCEDURE MOEmean(IN replicationNumber : INTEGER;
                 IN currentSample     : REAL);

VAR
    newMOE, oldAvg : REAL;

BEGIN
    newMOE := currentSample / FLOAT(TotalOPFORstarting);
    oldAvg := MeanMOE;
    MeanMOE := ((FLOAT(replicationNumber) * oldAvg) +
                newMOE) / FLOAT(replicationNumber + 1);
    MOEvariance(replicationNumber, oldAvg, newMOE);
END PROCEDURE;

PROCEDURE MOEvariance(IN replicationNumber : INTEGER;
                     IN oldAvg            : REAL;
                     IN currentSample     : REAL);

VAR
    rn,
    oldVariance : REAL;

BEGIN
    rn := FLOAT(replicationNumber);
    oldVariance := VarianceMOE;
    IF replicationNumber = 0
        VarianceMOE := 0.0;
    ELSIF replicationNumber = 1
        VarianceMOE := POWER((oldAvg - currentSample), 2.0) / 2.0;
    ELSE
        VarianceMOE := (((rn - 1.0) / rn) * oldVariance) +
                        POWER(oldAvg, 2.0) +
                        ((1.0 / rn) * POWER(currentSample, 2.0)) -
                        (((rn + 1.0) / rn) * POWER(MeanMOE, 2.0));
    END IF;
END PROCEDURE;

PROCEDURE ReportStats;
BEGIN
    ASK OutputFile TO WriteString("The mean Destroy MOE over ");

```

```

ASK OutputFile TO WriteInt(numberOfReplications,5);
ASK OutputFile TO WriteString(" replications is ");
ASK OutputFile TO WriteReal(MeanMOE,6,4);
ASK OutputFile TO WriteLn;
ASK OutputFile TO WriteLn;
ASK OutputFile TO WriteString("The variance of the Destroy MOE is ");
ASK OutputFile TO WriteReal(VarianceMOE,6,4);
ASK OutputFile TO WriteLn;
ASK OutputFile TO WriteLn;
ASK OutputFile TO WriteString("The mean mission time is ");
ASK OutputFile TO WriteReal(meanMissionTime,8,4);
ASK OutputFile TO WriteString(" hrs.");
ASK OutputFile TO WriteLn;
ASK OutputFile TO WriteLn;
ASK OutputFile TO WriteString("The mean attrition for the");
ASK OutputFile TO WriteString(" battalion was ");
ASK OutputFile TO WriteReal(percentAttrition * 100.0,8,4);
ASK OutputFile TO WriteString(" percent.");
END PROCEDURE;

END MODULE.

```

## L. MENU

DEFINITION MODULE Menu;

TYPE

PROCEDURE RunMenu1;  
PROCEDURE RunMenu2;  
PROCEDURE CleanUp;

VAR

selectedModel,  
numberOfReplications : INTEGER;  
replicating,  
walkingThru,  
playingArty : BOOLEAN;

END MODULE.

IMPLEMENTATION MODULE Menu;

```
FROM CRTMod  IMPORT ClearScreen;
FROM IOMod   IMPORT ReadKey;
FROM MapRecon IMPORT UnitRoute, UnitTargetList;
FROM OPFOR   IMPORT EnemyVehicleRef;
```

TYPE

PROCEDURE RunMenu1;

VAR

selection : CHAR;

BEGIN

ClearScreen;

OUTPUT; OUTPUT; OUTPUT; OUTPUT;

OUTPUT(" Welcome to the Light Infantry Attack Simulation ");

OUTPUT; OUTPUT;

OUTPUT(" Select the Tactic you wish to experiment with.");

OUTPUT; OUTPUT;

OUTPUT(" (1) Base-Line Model ");

OUTPUT;

OUTPUT(" (2) Rear Attack ");

OUTPUT;

OUTPUT(" (3) Flank Attack ");

OUTPUT;

selection := ReadKey();

selectedModel := STRTOINT(selection);

RunMenu2;

END PROCEDURE;

PROCEDURE RunMenu2;

VAR

selection : CHAR;

BEGIN

ClearScreen;

OUTPUT; OUTPUT; OUTPUT; OUTPUT; OUTPUT; OUTPUT;

OUTPUT(" You have the option to : ");

OUTPUT; OUTPUT; OUTPUT;

OUTPUT("(1) Replicate the model a fixed number of times, or...");

OUTPUT;

OUTPUT("(2) Conduct a model Walk-Through WITH Artillery, or");

OUTPUT;

OUTPUT("(3) Conduct a Walk-Through WITHOUT Artillery");

OUTPUT;

selection := ReadKey();

IF selection = "1"

replicating := TRUE;

walkingThru := FALSE;

playingArty := TRUE;

ELSIF selection = "2"

replicating := FALSE;

walkingThru := TRUE;

```

        numberOfReplications := 1;
        playingArty := TRUE;
    ELSE
        replicating := FALSE;
        walkingThru := TRUE;
        numberOfReplications := 1;
        playingArty := FALSE;
    END IF;
    IF replicating
        OUTPUT; OUTPUT; OUTPUT; OUTPUT; OUTPUT;
        OUTPUT(" Enter the number of replications");
        INPUT(numberOfReplications);
    END IF;
    ClearScreen;
END PROCEDURE;

PROCEDURE CleanUp;
BEGIN
    DISPOSE(EnemyVehicleRef);
    DISPOSE(UnitRoute);
    DISPOSE(UnitTargetList);
END PROCEDURE;

END MODULE.

```

## APPENDIX B. INPUT DATA FILES

### A. OPFOR DATA

Veh ID #	Type	Xref #	location	orientation
107	0		NK388180	135
140	0		NK361194	70
138	0		NK359190	80
141	0		NK358185	115
208	2		NK357177	97
94	0		NK350201	45
99	0		NK335188	110
108	0		NK332182	130
106	0		NK331180	130
162	3		NK329174	90
118	1		NK332157	80
95	0		NK329190	120
97	0		NK326189	135
113	0		NK315181	108

### B. MISSILE DATA

Missile	Vel	MaxEffRange	Prep	Acquire	Cut	AmmoCount	pHit
DRAGON							
66.667		1000.0	20.0	5.0	2.0	2	0.6475
TOW (HMMWV mounted)							
178.571		3750.0	5.0	5.0	2.0	10	0.7322

### C. P-KILL DATA

DRAGON vs.	front	flank	rear
BMP	0.72	0.655	0.72
BRDM	0.99	0.985	0.99
T72	0.185	0.405	0.185
ZSU234	0.80	0.815	0.80
TOW vs.	front	flank	rear
BMP	0.965	0.965	0.965
BRDM	0.995	0.985	0.995



T72	0.225	0.475	0.225
ZSU234	0.965	0.965	0.965

#### D. TRANSPORTATION DATA

METHOD	DAY	NIGHT	Conv Factor(m/s)
Foot	2.4	1.6	0.2778
Truck	12.0	8.0	0.2778
AirAssault	145.0	100.0	0.5111

## APPENDIX C. SCENARIO INPUT

### A. BASELINE MODEL

Symbol Xref#	coordinate	# firing pos	Firing Psns		
A Company					
14	NK325185	0			
13	NK333175	3	NK332172	NK333173	NK335176
12	NK336171	3	NK334170	NK335172	NK337173
10	NK344164	0			
9	NK360164	0			
8	NK369171	0			
7	NK372181	0			
5	NK375188	0			
3	NK390198	0			
2	NK405199	0			
1	NK411200	0			
0	NK417198	0			
A Company Target List					
# targets	Target ID numbers				
4	162 106 108 113				
B Company					
14	NK330195	0			
13	NK335185	3	NK334184	NK335185	NK337187
12	NK341183	3	NK339180	NK341183	NK343184
10	NK344164	0			
9	NK360164	0			
8	NK369171	0			
7	NK372181	0			
5	NK375188	0			
3	NK390198	0			
2	NK405199	0			
1	NK411200	0			
0	NK417198	0			
B Company Target List					
# targets	Target ID numbers				
3	97 95 99				
C Company					
14	NK360187	0			
13	NK362188	3	NK360185	NK362188	NK362191
12	NK365191	3	NK364188	NK364191	NK364192
6	NK368196	0			
4	NK369204	0			
1	NK360201	0			
0	NK353203	0			
C Company Target List					
# targets	Target ID numbers				
4	140 138 141 208				

## B. REAR ATTACK MODEL

Symbol Xref#	coordinate	# firing pos	Firing Psns
A Company			
14	NK331176	0	
13	NK322179	3	NK323180 NK323178 NK322177
12	NK315185	3	NK315183 NK316184 NK316185
3	NK311183	0	
2	NK303188	0	
1	NK301190	0	
0	NK297193	0	
A Company Target List			
# targets		Target ID numbers	
4		113	162 106 108
B Company			
14	NK332186	0	
13	NK326189	3	NK324186 NK326189 NK327191
12	NK320190	3	NK319189 NK320190 NK321191
4	NK315193	0	
1	NK301190	0	
0	NK297193	0	
B Company Target List			
# targets		Target ID numbers	
3		95	97 99
C Company			
14	NK357180	0	
13	NK356187	3	NK354184 NK356187 NK357188
12	NK351192	3	NK349189 NK351192 NK353194
7	NK346195	0	
6	NK335198	0	
5	NK318200	0	
1	NK301190	0	
0	NK297193	0	
C Company Target List			
# targets		Target ID numbers	
5		94	140 138 141 208

# C. FLANK ATTACK MODEL

Symbol Xref#	coordinate	# firing pos	Firing Psns		
A Company					
14	NK329178	0			
13	NK325188	3	NK322187	NK325188	NK323187
12	NK326199	3	NK325199	NK326199	NK327199
2	NK329210	0			
1	NK335217	0			
0	NK335219	0			
A Company Target List					
# targets		Target ID numbers			
3		95 97 113			
B Company					
14	NK331178	0			
13	NK331190	3	NK329184	NK330182	NK332185
12	NK333198	3	NK332198	NK333198	NK334198
3	NK333207	0			
1	NK335217	0			
0	NK335219	0			
B Company Target List					
# targets		Target ID numbers			
4		99 108 106 162			
C Company					
14	NK355180	0			
13	NK356185	3	NK354185	NK356185	NK359186
12	NK360198	3	NK355199	NK360198	NK361198
5	NK354204	0			
4	NK345207	0			
1	NK335217	0			
0	NK335219	0			
C Company Target List					
# targets		Target ID numbers			
5		94 140 138 141 208			

#### APPENDIX D. SAMPLE ATTRITION OUTPUT

Attrition to platoon C1 with 13 losses.

Strengths for each class of soldier

rifleman	8
autorifleman	3
grenadier	3
machinegunner	0
dragongunner	1
leader	11

Attrition to platoon C2 with 13 losses.

Strengths for each class of soldier

rifleman	7
autorifleman	4
grenadier	3
machinegunner	1
dragongunner	2
leader	9

Attrition to platoon C3 with 13 losses.

Strengths for each class of soldier

rifleman	8
autorifleman	3
grenadier	5
machinegunner	1
dragongunner	1
leader	8

Attrition to platoon A1 with 8 losses.

Strengths for each class of soldier

rifleman	10
autorifleman	3
grenadier	6
machinegunner	1
dragongunner	1
leader	10

Attrition to platoon A2 with 8 losses.

Strengths for each class of soldier

rifleman	9
autorifleman	3
grenadier	5
machinegunner	1
dragongunner	2
leader	11

Attrition to platoon A3 with 8 losses.

Strengths for each class of soldier

rifleman	5
autorifleman	5
grenadier	6
machinegunner	1
dragongunner	2
leader	12

Attrition to platoon B1 with 17 losses.

Strengths for each class of soldier

rifleman	6
----------	---

autorifleman 4  
grenadier 2  
machinegunner 1  
dragongunner 0  
leader 9

Attrition to platoon B2 with 17 losses.

Strengths for each class of soldier

rifleman 5  
autorifleman 3  
grenadier 3  
machinegunner 1  
dragongunner 1  
leader 9

Attrition to platoon B3 with 17 losses.

Strengths for each class of soldier

rifleman 6  
autorifleman 3  
grenadier 4  
machinegunner 1  
dragongunner 1  
leader 7

## APPENDIX E. SAMPLE ENGAGEMENT HISTORY

A11 damaged 113  
B11 killed 97  
C31 missed 94  
C32 killed 140  
C21 damaged 138  
B21 killed 95  
C11 missed 141  
B31 missed 99  
C31 moved out of range of target 94  
C11 damaged 141  
C21 killed 138  
C12 damaged 208  
B31 killed 99  
C12 killed 208  
Handing over 141 to platoon C2  
C22 killed 141  
A22 missed 162  
A11 killed 113  
A31 damaged 106  
A21 damaged 108  
A22 damaged 162  
A31 missed 106  
A21 missed 108  
Reassigning 106 to A32  
A32 damaged 106  
A32 killed 106  
Handing over 162 to platoon A1  
A12 missed 162  
A12 missed 162  
Unable to handover target 162  
Unable to handover target 108

## APPENDIX F. BASELINE MODEL OUTPUT

### A. RESULTS OF 500 REPLICATIONS

The mean Destroy MOE over 500 replications is 0.5881

The variance of the Destroy MOE is 0.0105

The mean mission time is 8.0390 hrs.

The mean attrition for the battalion was 33.9972 percent.

### B. RESULTS OF A TRIAL WITHOUT ARTILLERY

Enemy BMP number 138 KIA.

Killed at H + 7.3hrs by weapon type dragon from a frontal shot.

Enemy BMP number 141 KIA.

Killed at H + 7.3hrs by weapon type dragon from a flank shot.

Enemy BMP number 99 KIA.

Killed at H + 7.3hrs by weapon type dragon from a frontal shot.

Enemy BMP number 97 KIA.

Killed at H + 7.9hrs by weapon type dragon from a frontal shot.

Enemy T72 number 208 KIA.

Killed at H + 7.9hrs by weapon type dragon from a flank shot.

Enemy BMP number 95 KIA.

Killed at H + 8.0hrs by weapon type dragon from a frontal shot.

Enemy BMP number 140 KIA.

Killed at H + 8.0hrs by weapon type dragon from a flank shot.

Enemy BMP number 108 KIA.

Killed at H + 8.0hrs by weapon type dragon from a frontal shot.

Enemy ZSU234 number 162 KIA.

Killed at H + 8.2hrs by weapon type dragon from a frontal shot.

The mean Destroy MOE over 1 replications is 0.6429

The variance of the Destroy MOE is 0.0000

The mean mission time is 8.2474 hrs.



### C. RESULTS OF A TRIAL WITH ARTILLERY

Enemy BMP number 138 KIA.

Killed at H + 7.3hrs by weapon type dragon from a frontal shot.

Enemy BMP number 141 KIA.

Killed at H + 7.3hrs by weapon type dragon from a flank shot.

Enemy BMP number 99 KIA.

Killed at H + 7.3hrs by weapon type dragon from a frontal shot.

Enemy BMP number 140 KIA.

Killed at H + 8.0hrs by weapon type dragon from a flank shot.

Enemy ZSU234 number 162 KIA.

Killed at H + 8.0hrs by weapon type dragon from a frontal shot.

Enemy BMP number 108 KIA.

Killed at H + 8.0hrs by weapon type dragon from a frontal shot.

Enemy BMP number 106 KIA.

Killed at H + 8.1hrs by weapon type dragon from a frontal shot.

The mean Destroy MOE over 1 replications is 0.5000

The variance of the Destroy MOE is 0.0000

The mean mission time is 8.4447 hrs.

The mean attrition for the battalion was 34.3475 percent.

## APPENDIX G. REAR ATTACK MODEL OUTPUT

### A. RESULTS OF 500 REPLICATIONS

The mean Destroy MOE over 500 replications is 0.6757

The variance of the Destroy MOE is 0.0158

The mean mission time is 5.1685 hrs.

The mean attrition for the battalion was 33.9972 percent.

### B. RESULTS OF A TRIAL WITHOUT ARTILLERY

Enemy BMP number 97 KIA.

Killed at H + 4.1hrs by weapon type dragon from a rear shot.

Enemy BMP number 140 KIA.

Killed at H + 4.1hrs by weapon type dragon from a rear shot.

Enemy BMP number 95 KIA.

Killed at H + 4.1hrs by weapon type dragon from a rear shot.

Enemy BMP number 138 KIA.

Killed at H + 4.9hrs by weapon type dragon from a rear shot.

Enemy BMP number 99 KIA.

Killed at H + 4.9hrs by weapon type dragon from a rear shot.

Enemy T72 number 208 KIA.

Killed at H + 4.9hrs by weapon type dragon from a flank shot.

Enemy BMP number 141 KIA.

Killed at H + 5.0hrs by weapon type dragon from a rear shot.

Enemy BMP number 113 KIA.

Killed at H + 5.0hrs by weapon type dragon from a frontal shot.

Enemy BMP number 106 KIA.

Killed at H + 5.1hrs by weapon type dragon from a flank shot.

The mean Destroy MOE over 1 replications is 0.6429

The variance of the Destroy MOE is 0.0000

The mean mission time is 5.1876 hrs.

### C. RESULTS OF A TRIAL WITH ARTILLERY

Enemy BMP number 94 KIA.

Killed at H + 4.1hrs by weapon type dragon from a flank shot.

Enemy BMP number 95 KIA.

Killed at H + 4.1hrs by weapon type dragon from a rear shot.

Enemy BMP number 140 KIA.

Killed at H + 4.2hrs by weapon type dragon from a rear shot.

Enemy BMP number 99 KIA.

Killed at H + 4.9hrs by weapon type dragon from a rear shot.

Enemy BMP number 138 KIA.

Killed at H + 5.0hrs by weapon type dragon from a rear shot.

Enemy BMP number 97 KIA.

Killed at H + 5.0hrs by weapon type dragon from a rear shot.

Enemy ZSU234 number 162 KIA.

Killed at H + 5.0hrs by weapon type dragon from a rear shot.

Enemy BMP number 106 KIA.

Killed at H + 5.0hrs by weapon type dragon from a flank shot.

The mean Destroy MOE over 1 replications is 0.5714

The variance of the Destroy MOE is 0.0000

The mean mission time is 5:3100 hrs.

The mean attrition for the battalion was 34.3475 percent.

## APPENDIX H. FLANK ATTACK MODEL OUTPUT

### A. RESULTS OF 500 REPLICATIONS

The Mean Destroy MOE over 500 replications is 0.6330

The variance of the Destroy MOE is 0.0196

The mean mission time is 3.9922 hrs.

The mean attrition for the battalion was 33.9972 percent.

### B. RESULTS OF A TRIAL RUN WITHOUT ARTILLERY

Enemy BMP number 94 KIA.

Killed at H + 2.5hrs by weapon type dragon from a flank shot.

Enemy BMP number 95 KIA.

Killed at H + 2.5hrs by weapon type dragon from a flank shot.

Enemy BMP number 97 KIA.

Killed at H + 3.6hrs by weapon type dragon from a flank shot.

Enemy BMP number 138 KIA.

Killed at H + 3.7hrs by weapon type dragon from a flank shot.

Enemy T72 number 208 KIA.

Killed at H + 3.7hrs by weapon type dragon from a flank shot.

Enemy BMP number 140 KIA.

Killed at H + 3.7hrs by weapon type dragon from a flank shot.

Enemy BMP number 141 KIA.

Killed at H + 3.8hrs by weapon type dragon from a rear shot.

Enemy BMP number 99 KIA.

Killed at H + 4.3hrs by weapon type dragon from a flank shot.

Enemy BMP number 108 KIA.

Killed at H + 4.3hrs by weapon type dragon from a rear shot.

Enemy BMP number 106 KIA.

Killed at H + 4.4hrs by weapon type dragon from a rear shot.

Enemy ZSU234 number 162 KIA.

Killed at H + 4.6hrs by weapon type dragon from a flank shot.

The mean Destroy MOE over 1 replications is 0.7857

The variance of the Destroy MOE is 0.0000

The mean mission time is 4.5680 hrs.

### C. RESULTS OF A TRIAL RUN WITH ARTILLERY

Enemy BMP number 94 KIA.

Killed at H + 2.6hrs by weapon type dragon from a flank shot.

Enemy BMP number 97 KIA.

Killed at H + 2.6hrs by weapon type dragon from a rear shot.

Enemy BMP number 95 KIA.

Killed at H + 3.6hrs by weapon type dragon from a flank shot.

Enemy BMP number 138 KIA.

Killed at H + 3.7hrs by weapon type dragon from a rear shot.

Enemy BMP number 140 KIA.

Killed at H + 3.7hrs by weapon type dragon from a flank shot.

Enemy BMP number 141 KIA.

Killed at H + 3.8hrs by weapon type dragon from a rear shot.

Enemy BMP number 113 KIA.

Killed at H + 3.8hrs by weapon type dragon from a flank shot.

The mean Destroy MOE over 1 replications is 0.5000

The variance of the Destroy MOE is 0.0000

The mean mission time is 4.3956 hrs.

The mean attrition for the battalion was 34.3475 percent.

## LIST OF REFERENCES

1. Vuono, Carl E., *The United States Army - A Strategic Force for the 1990s and Beyond*, Department of the Army, January 1990, p. 3.
2. US Department of the Army Field Manual (FM) 100-5, *Operations*, Government Printing Office, Washington, DC, May 1986.
3. Boylan, Peter J., "Complementary Force Operations," *Military Review*, v. LXX, no. 6, June 1990.
4. US Department of the Army Field Manual (FM) 7-20, *The Infantry Battalion (Infantry, Airborne, Air Assault, Ranger)*, p. 1-2, Government Printing Office, Washington, DC, 3 April 1978.
5. US Department of the Army Field Manual (FM) 101-5-1, *Operational Terms and Symbols*, Government Printing Office, Washington, DC, 21 October 1985.
6. Cronin, Patrick M., "Clausewitz Condensed," *Military Strategy: Theory and Application*, US Army War College, Carlisle Barracks, PA, 1989, pp. 85-89.
7. CACI Products Company, Reference Manual, *MODSIM II The Language for Object-Oriented Programming*, 1990.
8. Bratley, Paul, Fox, Bennett L., and Schrage, Linus E., *A Guide To Simulation*, 2d ed., Springer-Verlag, 1987.
9. US Department of the Army Field Manual (FM) 5-34, *Engineer Field Data*, Government Printing Office, Washington, DC, 14 September 1987, p. 1-3.
10. Naval Postgraduate School, *Notes on Firing Theory*, by Alan Washburn, pp. 13-14, 1985.

11. Telephone conversation between CPT Alicea, US Army, National Training Center, and CPT Bundy, US Army, TRAC Monterey, 17 May 1991.
12. Law, Averill M., and Kelton, David W., *Simulation Modeling and Analysis*, McGraw-Hill, 1982.
13. Take Home Package, AA89xxxx, US Army National Training Center, 1988.
14. User's Documentation , *General-purpose NTC Analysis Training Tool*, Version 1.1, Army Research Institute--Presidio of Monterey, 15 November 1989.

## INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Deputy Undersecretary of the Army for Operations Research Room 2E261, The Pentagon Washington, D.C. 20310	1
4.	Director Attn: Mr. E.B. Vandiver III U.S. Army Concepts Analysis Agency Bethesda, MD 20814	1
5.	Commander and Director U.S. Army TRADOC Analysis Command TRAC-FLVN Attn: ATRC-ZD (Mr. Bauman) Fort Leavenworth, KS 66027-5200	1
6.	Commander and Director U.S. Army TRADOC Analysis Command TRAC-WSMR Attn: ATRC-W (Dr. Collier) White Sands Missile Range, NM 88002-5502	1
7.	Professor Samuel H. Parry Department of Operations Research Naval Postgraduate School, Code 55Py Monterey, CA 93943-5000	2
8.	Professor Michael P. Bailey Department of Operations Research Naval Postgraduate School, Code OR/Ba Monterey, CA 93943-5000	3
9.	Commander and Director U.S. Army TRADOC Analysis Command TRAC-Monterey Monterey, CA 93943	1



10. Captain Steven J. Hutchison  
Department of Mathematical Sciences  
United States Military Academy  
West Point, NY 10996

1